⌐computational complexity

PARALLEL APPROXIMATION OF MIN-MAX PROBLEMS

Gus Gutoski and Xiaodi Wu

Abstract. This paper presents an efficient parallel approximation scheme for a new class of min-max problems. The algorithm is derived from the matrix multiplicative weights update method and can be used to find near-optimal strategies for competitive two-party classical or quantum interactions in which a referee exchanges any number of messages with one party followed by any number of additional messages with the other. It considerably extends the class of interactions which admit parallel solutions, demonstrating for the first time the existence of a parallel algorithm for an interaction in which one party reacts adaptively to the other.

As a consequence, we prove that several competing-provers complexity classes collapse to PSPACE, such as QRG(2), SQG and two new classes called DIP and DQIP. A special case of our result is a parallel approximation scheme for a specific class of semidefinite programs whose feasible region consists of lists of semidefinite matrices that satisfy a transcript-like consistency condition. Applied to this special case, our algorithm yields a direct polynomial-space simulation of multi-message quantum interactive proofs resulting in a first-principles proof of QIP = PSPACE.

Keywords. Parallel approximation algorithm, semidefinite programming, zero-sum games, quantum interactive proofs with competing provers.

Subject classification. 68Q05, 68Q10, 68Q12, 68Q15, 68W25, 68W10, 90C22, 90C47.

1. Introduction

This paper presents a parallel approximation scheme for a new class of min-max problems with applications to classical and quantum zero-sum games and interactive proofs. In order to describe this class of min-max problems, let us begin by considering a semidefinite program (SDP) of the form

(1.1)
$$
\begin{aligned}
\text{minimize} \quad & \mathrm{Tr}(X_k P) \\
\text{subject to} \quad & \mathrm{Tr}_{\mathbb{M}_n}(X_{i+1}) = \Phi_i(X_i) \text{ for } i = 1, \ldots, k-1, \\
& \mathrm{Tr}(X_1) = 1, \\
& 0 \preceq X_1, \ldots, X_k \in \mathbb{M}_{mn}.
\end{aligned}
$$

Here, $\mathbb{M}_d$ denotes the space of all $d \times d$ complex matrices, and $\mathrm{Tr}_{\mathbb{M}_n}$ is the *partial trace*—the unique linear map from matrices to matrices satisfying

$$
\mathrm{Tr}_{\mathbb{M}_n} : \mathbb{M}_{mn} \to \mathbb{M}_m : A \otimes B \mapsto \mathrm{Tr}(B)A
$$

for every choice of $A \in \mathbb{M}_m$ and $B \in \mathbb{M}_n$. An SDP (1.1) is specified by arbitrary choices of a positive semidefinite matrix $P \in \mathbb{M}_{mn}$ with $\|P\| \leq 1$ and completely positive and trace-preserving linear maps $\Phi_1, \ldots, \Phi_{k-1} : \mathbb{M}_{mn} \to \mathbb{M}_m$. (A linear map Φ is *positive* if $\Phi(X) \succeq 0$ whenever $X \succeq 0$. Such a map is *completely positive* if $\Phi \otimes \mathbb{1}_{\mathbb{M}_d}$ is positive for every positive integer d.)

Let $\mathbf{A}$ denote the feasible region of the SDP (1.1) (which is always nonempty), and let $\mathbf{P} \subset \mathbb{M}_{mn}$ be a nonempty compact convex subset of positive semidefinite matrices having operator norm at most 1. We are concerned with the following min-max problem, which is a generalization of the SDP (1.1):

(1.2)
$$
\lambda(\mathbf{A}, \mathbf{P}) \stackrel{\text{def}}{=} \min_{(X_1, \ldots, X_k) \in \mathbf{A}} \ \max_{P \in \mathbf{P}} \ \mathrm{Tr}(X_k P).
$$

The ordering of minimization and maximization is immaterial, as implied by well-known extensions of von Neumann's Min-Max Theorem (Fan 1953; von Neumann 1928) given the fact that $\mathbf{A}, \mathbf{P}$ are convex compact sets and $\mathrm{Tr}(X_k P)$ is a bilinear form over the two sets.

Our main result is an efficient parallel oracle-algorithm for finding approximate solutions to the min-max problem (1.2) and for approximating the quantity $\lambda(\mathbf{A}, \mathbf{P})$ given an oracle for optimization over the set $\mathbf{P}$. We also describe parallel implementations of this oracle for certain sets $\mathbf{P}$, yielding an unconditionally efficient parallel approximation scheme for the min-max problem (1.2) for those choices of $\mathbf{P}$. This result is stated formally below as Theorem 1.5. Before stating this theorem, let us clarify terminology.

1.1. Review of parallel computation: formal statement of results. Recall that a *parallel algorithm* is described by a family of logarithmic-space uniform Boolean circuits. The uniformity constraint insures that the size of each circuit in the family scales as a polynomial in the bit length of the input, and therefore, the family represents a polynomial-time computation. Boolean circuits are an ideal model of parallel computation because computational activity can occur concurrently at many different gates in the circuit. Indeed, the run time of a parallel algorithm is determined by the *depth* of its circuits, which might be much smaller than the total *size* of its circuits.

A parallel algorithm is said to be *efficient* if the depth of its circuits (and therefore the run time of the algorithm) scales as a polynomial in the *logarithm* of the bit length of the input. The complexity class NC consists of those functions which can be computed by efficient parallel algorithms. Efficient parallel algorithms are sometimes called "NC algorithms" or "NC computations." The reader is referred to Papadimitriou (1994) for an accessible introduction to parallel computation.

An *oracle-algorithm* is an algorithm endowed with the ability to get instantaneous answers to questions that fall within the scope of some specific *oracle*. In our case, we assume an oracle for optimization over $\mathbf{P}$, which instantly solves problems of the form

PROBLEM 1.3 (Optimization over $\mathbf{P}$).

Input: *A matrix $X \succeq 0$ with $\mathrm{Tr}(X) = 1$ and an accuracy parameter $\delta > 0$.*
Output: *A near-optimal element $P^\star \in \mathbf{P}$ such that $\mathrm{Tr}(XP^\star) \geq \mathrm{Tr}(XP) - \delta$ for all $P \in \mathbf{P}$.*

An oracle is incorporated into the circuit model of computation by supplementing a standard gate set (such as $\{\mathrm{AND}, \mathrm{OR}, \mathrm{NOT}\}$) with a special *oracle gate*. This oracle gate has many input bits (describing the question) and many output bits (describing the answer). As with standard gates, each oracle gate contributes unit cost to circuit size and run time.

An *approximation scheme* refers to an algorithm that computes one or more quantities to a given precision δ and whose run time is efficient for each fixed choice of $\delta > 0$ but does not necessarily scale well with δ. In the circuit model (and other models too), this property is encapsulated by defining the underlying problem so that the accuracy parameter $\delta = 1/s$ is specified in unary as 1^s, thus forcing the bit length of the input to be proportional to $1/\delta$ instead of $1/\log(\delta)$. The choice to specify the accuracy parameter in unary allows parallel approximation schemes to be described neatly by log-space uniform circuits with polylog depth.

The following is a formal statement of the problem solved by our algorithm.

PROBLEM 1.4 (Approximation of $\lambda(\mathbf{A}, \mathbf{P})$).

Input: *Completely positive and trace-preserving linear maps $\Phi_1, \ldots, \Phi_{k-1}$ specifying the feasible region $\mathbf{A}$ of an SDP of the form (1.1). An accuracy parameter $\delta > 0$.*

Oracle: *Optimization over $\mathbf{P}$ (Problem 1.3).*

Output: *Near-optimal elements $(X_1^\star, \ldots, X_k^\star) \in \mathbf{A}$ and $P^\star \in \mathbf{P}$ such that*

$$\mathrm{Tr}(X_k^\star P) \leq \lambda(\mathbf{A}, \mathbf{P}) + \delta \text{ for all } P \in \mathbf{P},$$
$$\mathrm{Tr}(X_k P^\star) \geq \lambda(\mathbf{A}, \mathbf{P}) - \delta \text{ for all } (X_1, \ldots, X_k) \in \mathbf{A},$$

and a quantity $\tilde{\lambda}$ with $|\tilde{\lambda} - \lambda(\mathbf{A}, \mathbf{P})| \leq \delta$.

The maps $\Phi_1, \ldots, \Phi_{k-1}$ are linear maps from a complex vector space of dimension $(mn)^2$ to another complex space of dimension m^2. As such, these maps can be represented by complex matrices of size $m^2 \times (mn)^2$. In both Problem 1.3 and Problem 1.4, it is assumed that the real and imaginary parts of each entry in each input matrix are represented as rational numbers expressed

as the ratio of two p-bit integers written in binary for some p that is promised to scale as a polynomial in the dimension mn. (Indeed, it suffices for our purpose that p scales *logarithmically* with mn.) As suggested previously, it is also assumed that the accuracy parameter δ is represented in unary. These assumptions allow us to focus on the quantities $mn, k, 1/\delta$ as the dominating factors determining the run time of our parallel algorithm. We may now state our main result.

THEOREM 1.5 (Main result). *There is a parallel oracle-algorithm for Problem 1.4 (Approximation of $\lambda(\mathbf{A}, \mathbf{P})$) with run time bounded by a polynomial in k, $1/\delta$, and $\log(mn)$. This algorithm is efficient if $k, 1/\delta$ are promised to scale as a polynomial in $\log(mn)$.*

1.2. Application: parallel approximation of semidefinite programs. The SDP (1.1) is recovered from (1.2) in the special case where $\mathbf{P} = \{P\}$ is a singleton set. Thus, a special case of Theorem 1.5 is a parallel approximation scheme for SDPs of the form (1.1).

We restricted attention to SDPs for which $\|P\|, \mathrm{Tr}(X_1) \leq 1$ because this restriction does not interfere with our application to quantum interactive protocols, and because the run time of our parallel algorithm scales polynomially with the largest eigenvalue of P and with the trace of X_1, so it is only efficient when these quantities are bounded by a fixed polynomial in the logarithm of the bit length of the input $P, \Phi_1, \ldots, \Phi_{k-1}$. (In keeping with convention, one can think of these quantities as the *width* of the SDPs we consider. Our algorithm is efficient only for *width-bounded* SDPs.)

It has long since been known that the problem of approximating the optimal value of an arbitrary SDP is logspace-hard for P (Megiddo 1992; Serna 1991), so there cannot be a parallel approximation scheme for *all* SDPs unless NC = P. The precise extent to which SDPs admit parallel solutions is not known. This special case of our result adds considerably to the set of such SDPs, subsuming all prior work in the area at the time it was made public. (Since that time, parallel approximation schemes have been found for some SDPs of unbounded width that are not covered by our scheme (Jain & Yao 2011, 2012; Peng & Tangwongsan 2012).)

Some of what is known about SDPs in this respect is inherited knowledge from linear programs (LPs). For example, Luby & Nisan (1993) describe a parallel approximation scheme for so-called *positive* LPs of the form

minimize xp^* subject to $Cx \geq q$ and $x \geq 0$,

where each entry of the matrix C and vectors p, q is a nonnegative real number. Young (2001) provides a generalization of Luby-Nisan to arbitrary mixed packing and covering problems. By contrast, Trevisan & Xhafa (1998) show that it is P-hard to find *exact* solutions for positive LPs.[1]

The notion of a positive instance of an LP can be generalized to SDPs as follows. An SDP of the form

minimize $\mathrm{Tr}(XP)$ subject to $\Psi(X) \succeq Q$ and $X \succeq 0$

is said to be *positive* if $P, Q \succeq 0$ and Ψ is a positive map. Of course, P-hardness of exact solutions for positive LPs implies P-hardness of exact solutions for positive SDPs. Jain & Watrous (2009) give a parallel approximation scheme for width-bounded positive SDPs. Subsequent improvements extend to all positive SDPs (Jain & Yao 2011; Peng & Tangwongsan 2012) and even to mixed packing and covering SDPs (Jain & Yao 2012).

The Jain-Watrous algorithm for positive SDPs is derived from a correspondence between positive SDPs and one-turn quantum games and can therefore be recovered as a special case of the work of the present paper. In their proof of QIP = PSPACE, Jain *et al.* (2011) give a parallel algorithm for a specific SDP based on quantum interactive proofs. It is not difficult to see that their SDP can be written in the form (1.1) considered in the present paper.

As mentioned above, our algorithm is not efficient when used for SDPs of unbounded width, leaving the recent works of Jain & Yao (2011, 2012) and Peng & Tangwongsan (2012) on mixed packing

---

[1] For clarification, a polynomial-time algorithm finds an *exact* solution to an LP or SDP if it finds solutions that are within ε of optimal in time polynomial in the bit length of ε—that is, $\log(1/\varepsilon)$. By contrast, an *approximation scheme* for LPs or SDPs finds solutions that are within ε of optimal with run time that depends super-polynomially in the bit length of ε—typically $1/\varepsilon$.

and covering SDPs as the only known parallel SDP approximation schemes that are not subsumed by the present work. These recent works do not subsume our results, as neither the SDP instance used in Jain *et al.* (2011) to prove QIP = PSPACE nor its generalization (1.1) in the present paper are mixed packing and covering SDPs.

1.3. Application: interactive proofs with competing provers.

1.3.1. Definitions.   An *interactive proof with competing provers* consists of a conversation between a *verifier* and two *provers* regarding some input string x. The verifier may use randomness, but must run in time that scales as a polynomial in the input length $|x|$; the provers are permitted unlimited computational power. One of the provers—the *yes-prover*—tries to convince the verifier to accept x, while the other—the *no-prover*—tries to convince the verifier to reject x. A decision problem L is said to admit an interactive proof with competing provers with *completeness c* and *soundness s* if there exists c, s with $c > s$ and a randomized polynomial-time verifier who meets the following conditions:

Completeness condition. If x is a yes-instance of L, then the yes-prover can convince the verifier to accept with probability at least c regardless of the no-prover's strategy.

Soundness condition. If x is a no-instance of L, then the no-prover can convince the verifier to reject with probability at least $1 - s$ regardless of the yes-prover's strategy.

The completeness and soundness parameters c, s need not be fixed constants, but may instead vary as a function of the input length $|x|$. If these parameters are not specified, then it is assumed that L admits an interactive proof with competing provers for some choice of $c(|x|), s(|x|)$ for which there exists a polynomial-bounded function $p(|x|)$ such that $c - s \geq 1/p$. The complexity class RG consists of all decision problems that admit interactive proofs with competing provers. (The acronym RG stands for "refereed games," a term inspired by the field of game theory).

Often in the study of interactive proofs, the precise values of c, s are immaterial because sequential repetition (or sometimes parallel

repetition) can be used to transform any verifier for which $c - s \geq 1/p$ into another verifier for which c tends toward one and s tends toward zero exponentially quickly in the bit length of x. (For example, sequential repetition followed by a majority of vote can be used to reduce the error for RG.) For this reason, it is typical to assume without loss of generality that c, s are constants such as $2/3, 1/3$ or that c is exponentially close to one and s is exponentially close to zero whenever it is convenient to do so. However, it is not always clear that a given complexity class is robust with respect to the choice of c, s so it is a good practice to be as inclusive as possible when defining these classes.

Interesting subclasses of RG are obtained by placing restrictions upon the number and timing of messages in the interaction between the verifier and provers. In this paper, we introduce one such subclass based upon interactions of the following form:

1. The verifier exchanges several messages with only the yes-prover.

2. After processing this interaction with the yes-prover, the verifier exchanges several additional messages with only the no-prover.

3. After further processing, the referee declares acceptance or rejection.

Interactive proofs of this form shall be called *double interactive proofs*: The verifier in such a protocol executes a standard single-prover interactive proof with the yes-prover followed by a second single-prover interactive proof with the no-prover. The class of problems that admit double interactive proofs shall be called DIP.

By contrast to RG, it is not immediately clear that the definition of DIP is robust with respect to the choice of parameters c, s. But it follows from our result that DIP is, in fact, robust with respect to the choice of c, s. Also, whereas RG is trivially closed under complement, the protocol for double interactive proofs is asymmetric, and so, it is not immediately clear that DIP is closed under complement. Again, it follows from our result that DIP is closed under complement.

Another example of an interesting subclass of RG is the family of *bounded-turn* classes. For each positive integer k, the class

RG(k) consists of those problems that admit an interactive proof with competing provers in which the verifier exchanges no more than k messages with each prover. It is understood that messages are exchanged with the provers in parallel so that RG(k), like RG, is trivially closed under complement.

Quantum interactive proofs with competing provers are defined similarly except that the verifier is a polynomial-time quantum computer who exchanges quantum information with the provers. The analogous complexity classes are denoted QRG, DQIP, and QRG(k).

1.3.2. Prior work. As noted by Feige & Kilian (1997); Feigenbaum *et al.* (1995), the results of Koller & Megiddo (1992); Koller *et al.* (1994) imply that RG $\subseteq$ EXP. The reverse containment was proven by Feige & Kilian (1997), yielding the characterization RG = EXP. It was proven in Gutoski & Watrous (2007) that QRG $\subseteq$ EXP, from which one obtains

$$QRG = RG = EXP,$$

which is the competing-provers version of the well-known collapse QIP = IP = PSPACE for single-prover interactive proofs (Jain *et al.* 2011; Lund *et al.* 1992; Shamir 1992).

For bounded-turn classes, the results of Fortnow *et al.* (2008) tell us that RG(1) is essentially a randomized version of S_2^P. Feige & Kilian (1997) proved RG(2) = PSPACE.[2] For bounded-turn quantum classes, Jain & Watrous (2009) proved QRG(1) $\subseteq$ PSPACE. The complexity of QRG(2) is an open question of Jain *et al.* (2011) that is solved in the present paper. The exact complexity of RG(k) and QRG(k) for all other k is not known.

Bounded-turn double quantum interactive proofs have been studied previously under the name *short quantum games*; the associated complexity class has been called SQG. In an effort to unify

---

[2] The class we call RG(2) is called RG(1) by Feige & Kilian (1997). This conflict in notation stems from the fact that we measure the length of an interaction in *turns* (*i.e.* messages per prover), whereas those authors measure an interaction in *rounds* of messages. This switch of notation was instigated by Jain & Watrous (2009), who required a convenient symbol for one-turn interactions.

the notation, let $DQIP(k, l)$ denote the class consisting of problems that admit a double quantum interactive proof with competing provers in which the verifier exchanges no more than k messages with the yes-prover followed by no more than l messages with the no-prover. The class SQG was first defined in Gutoski & Watrous (2005) to be equal to $DQIP(1, 2)$, wherein it was shown that this class contains $QIP = DQIP(poly, 0)$. The importance of short quantum games has been diminished by the proof of QIP = PSPACE, as containment of QIP is no longer such a peculiar property. However, the containment of PSPACE inside $DQIP(1, 2)$ is still interesting, as it is not known whether PSPACE is contained in $DIP(1, 2)$, the classical version of this class.

1.3.3. Our contribution. As we explain in Section 5, the oracle-algorithm of Theorem 1.5—together with a parallel implementation of a suitably chosen oracle—implies that near-optimal strategies for the provers in a double quantum interactive proof can be computed efficiently in parallel. The following containment then follows from a standard argument (summarized in Section 5.4).

THEOREM 1.6. $DQIP \subseteq PSPACE$.

This containment, when combined with the trivial containments $IP \subseteq DIP \subseteq DQIP$ and the well-known fact that $PSPACE \subseteq IP$ (Lund *et al.* 1992; Shamir 1992), yields the following characterization.

COROLLARY 1.7. $DQIP = DIP = PSPACE$.

As a special case of Corollary 1.7, we obtain the solution to an open problem of Jain *et al.* (2011):

COROLLARY 1.8. $QRG(2) = PSPACE$.

Another special case of our result is a direct polynomial-space simulation of multi-message quantum interactive proofs, resulting in a first-principles proof of QIP = PSPACE.

COROLLARY 1.9. QIP = PSPACE *via direct polynomial-space simulation of multi-message quantum interactive proofs.*

By contrast, all other known proofs (Jain *et al.* 2011; Wu 2010) rely upon the fact that the verifier can be assumed to exchange only three messages with the prover (Kitaev & Watrous 2000). The original proof of Jain *et al.* (2011) also relies on the additional fact that the verifier's only message to the prover can be just a single classical coin flip (Marriott & Watrous 2005).

Of course, every other competing-provers complexity class whose protocol can be cast as a double interactive proof also collapses to PSPACE, such as the aforementioned class DQIP$(1, 2)$ based on short quantum games.

It follows from the collapse of DQIP and DIP to PSPACE that these classes are closed under complement and that they are robust with respect to the choice of parameters c, s. (Indeed, it may be assumed that $c = 1$ and $s \leq 2^{-q}$ for any desired polynomially bounded function $q(|x|)$—see Section 6.3.)

Prior to the present work, polynomial-space algorithms were known only for two-turn classical interactive proofs with competing provers (RG(2)), for one-turn quantum interactive proofs with competing provers (QRG(1)), and for single-prover quantum interactive proofs (QIP). Our result unifies and subsumes all of these algorithms. It also demonstrates for the first time the existence of a polynomial-space algorithm for a competing-prover interaction (classical or quantum) in which one prover reacts adaptively to the other.

Finally, our results illustrate a difference in the effect of public randomness between *single*-prover interactive proofs and *competing*-prover interactive proofs. Any classical interactive proof with single prover can be simulated by another *public-coin* interactive proof where the verifier's messages to the prover consist entirely of uniformly random bits and the verifier uses no other randomness (Goldwasser & Sipser 1989). Extending the notion of public-coin interaction to competing-prover interactions, it is easy to see that any such interaction with a public-coin verifier can be simulated by a double interactive proof.[3] We therefore have that the

---

[3] *Proof sketch:* As the verifiers' questions to each prover are uniformly random, they cannot depend on prior responses from the other prover and

public-coin version of RG is a subset of DIP, which we now know
is equal to PSPACE. Thus, by contrast to the single-prover case
where public-coin-IP = IP, in the competing-prover case, we es-
tablish the following.

COROLLARY 1.10. public-coin-RG $\neq$ RG *unless* PSPACE = EXP.

1.4. Summary of techniques.

1.4.1. The matrix multiplicative weights update method.
The parallel oracle-algorithm we exhibit in the proof of Theo-
rem 1.5 is an example of the *matrix multiplicative weights update
method (MMW)* as presented in Arora *et al.* (2005); Kale (2007);
Warmuth & Kuzmin (2006). We draw upon the valuable experi-
ence of recent applications of this method to parallel algorithms
for quantum complexity classes (Jain *et al.* 2011, 2009; Jain &
Watrous 2009; Wu 2010). We also make extensive use of effi-
cient parallel algorithms for various matrix manipulation tasks,
such as computing the singular value decomposition or exponen-
tial of a matrix. The reader is referred to von zur Gathen (1993)
for more detail on parallel algorithms for matrix operations and to
works of Jain *et al.* (2011, 2009) for discussion of the use of these
algorithms in parallel implementations of the matrix multiplicative
weights update method.

    In its unaltered form, the MMW can be used to solve min-max
problems over the domain of *density operators*—positive semidef-
inite matrices X with $\text{Tr}(X) = 1$. We introduce a new extension
to this method for min-max problems over the domain **A** defined
in the SDP (1.1)—a domain consisting of *k-tuples of density oper-
ators* lying within a *strict subspace* of the affine space associated
with k-tuples of density operators. The high-level approach of our
method is as follows:

Foonote 3 continued

can therefore be reordered so that all messages with one prover are exchanged
before any messages with the other.

1. **Extend the domain from a single density matrix to a k-tuple of density matrices.**
   This step is straightforward: The MMW can be applied without complication to all k density matrices at the same time. (Equivalently, k density matrices may be viewed as a single, larger, block-diagonal density matrix.)

2. **Restrict the domain to a strict subspace of k-tuples of density matrices.**
   This step is more difficult. It is accomplished by relaxing the problem so as to allow *all k*-tuples, with an additional *penalty term* to remove incentive for the players to use inconsistent transcripts.

3. **Round strategies in the relaxed problem to strategies in the original protocol.**
   For this step, one must prove a "rounding" theorem (Theorem 3.1), which establishes that near-optimal, fully admissible strategies can be obtained from near-optimal strategies in the unrestricted domain with penalty term.

1.4.2. Finding optimal strategies for the provers in a double quantum interactive proof.   In Section 5, we observe that the verifier in a double quantum interactive proof induces a min-max problem of the form (1.2) in which elements of **A** correspond to strategies for the yes-prover and elements of **P** correspond to strategies for the no-prover. Thus, the parallel oracle-algorithm of Theorem 1.5—together with a parallel implementation of the oracle for optimization over **P**—can be used to find optimal strategies for the provers in a double quantum interactive proof.

Our implementation of this oracle is itself a special case of the algorithm of Theorem 1.5, so that the overall algorithm employs the MMW method *twice* in a two-level recursive fashion. At the top level, the MMW is used to iteratively converge toward an optimal strategy for the yes-prover; at the bottom level, the MMW is used again to solve an SDP for "best responses" for the no-prover to a given strategy for the yes-prover.

The central challenge in using the MMW to find optimal strategies for parties in a quantum interaction is to find a representation

for strategies that are amenable to the MMW method. In Kitaev's *transcript* representation (Kitaev 2002), the actions of a prover in a double quantum interactive proof are represented by a list $X_1, \ldots, X_k$ of density matrices that satisfy a special consistency condition that is captured by the definition of the feasible region **A** of the SDP (1.1). Intuitively, these density matrices correspond to "snapshots" of the state of the verifier's qubits at various times during the interaction. (See Figure 5.2 on page 415.)

The key property of double quantum interactive proofs that we exploit is the ability to draw a "temporal line" in the interaction before which only the yes-prover acts and after which only the no-prover acts. Given a transcript $X_1, \ldots, X_k$ for the yes-prover, the actions of the no-prover can then be represented by another transcript $Y_1, \ldots, Y_\ell$. By optimizing over all such transcripts, one obtains an oracle for "best responses" for the no-prover to a given strategy of the yes-prover as required by the MMW method.

1.4.3. Comparison of methods for semidefinite programming.

In their proof of QIP = PSPACE, Jain *et al.* (2011) employ the MMW to solve a special SDP for quantum interactive proofs by making direct use of the primal-dual approach described in the thesis of Kale (2007). Subsequent parallel algorithms for positive SDPs (Jain & Yao 2011; Peng & Tangwongsan 2012) and for mixed packing and covering SDPs (Jain & Yao 2012) are matrix generalizations (also based on MMW) of existing algorithms for linear programs (Luby & Nisan 1993; Young 2001).

We do not use any of these approaches for solving SDPs. Instead, we use the MMW to solve a min-max problem as suggested by the algorithmic proof (also presented in Kale (2007)) of a min-max theorem for a simple class of zero-sum quantum games. By introducing a penalty term for inadmissible strategies, we are able to extend this algorithm to a much richer class of games beyond the one-turn games considered by Kale. We wish to stress that our parallel algorithm for SDPs arises as a *special case* of a more general min-max algorithm, whereas previous approaches for SDPs do not generalize to min-max problems in any obvious way.

1.4.4. Comparison of proofs of QIP = PSPACE.   Unlike the present paper, the original proof of QIP = PSPACE due to Jain *et al.* (2011) does not take advantage of the transcript representation for arbitrary multi-turn strategies. Instead, as mentioned earlier, those authors derive a special SDP by invoking several nontrivial facts about quantum interactive proofs. Admittedly, their SDP does bear a resemblance to Kitaev's transcript conditions, but this resemblance is only superficial and their solution applies only to a very restricted subset of transcripts. Indeed, their derivation breaks down without the assumption that the verifier sends only classical messages to the prover.

Previously, one of us (Wu 2010) presented a simplified proof of QIP = PSPACE that, like the work of the present paper, employs Kale's algorithmic min-max theorem (Kale 2007) instead of the primal-dual approach for SDPs that was used in the original proof by Jain *et al.* (2011). The QIP-completeness of the quantum circuit distinguishability problem (Rosgen & Watrous 2005) means that quantum interactive proofs can be decided by approximating the diamond norm of the difference between two quantum channels. Wu (2010) noticed that the diamond norm can be approximated in this special case by a direct application of Kale's algorithmic min-max theorem. His result did not require the penalization method introduced in the present paper nor an attendant rounding theorem.

1.4.5. The Bures angle.   Finally, it is noteworthy that the proof of our rounding theorem (Theorem 3.1) contains an interesting and nontrivial application of the Bures angle, which is a distance measure for quantum states that is defined in terms of the more familiar fidelity function.

Properties of the trace norm, which captures the physical distinguishability of quantum states, are sufficient for most needs in quantum information. When some property of the fidelity is also required, one uses the Fuchs-van de Graaf inequalities to convert between the trace norm and fidelity (Fuchs & van de Graaf 1999). (These inequalities are listed in Eq. (2.3) of Section 2.3.)

However, every such conversion incurs a quadratic slackening of relevant accuracy parameters. Our study calls for repeated

conversions, which would incur an unacceptable exponential slackening if done naively via Fuchs-van de Graaf. Instead, we make only a *single* conversion between the trace norm and the Bures angle and then repeatedly exploit the simultaneous properties of (i) the triangle inequality, (ii) contractivity under quantum channels, and (iii) preservation of subsystem fidelity.

Although conversion inequalities between the trace norm and Bures metric are implied by Fuchs-van de Graaf, to our knowledge, explicit conversion inequalities have not yet appeared in the published literature. The required inequalities are derived in the present paper (Proposition 2.4).

2. Preliminaries

Hereafter, we must assume familiarity with standard concepts from quantum information, though we have attempted to minimize our use of quantum formalism for the benefit of a wider audience. The reader is referred to Nielsen & Chuang (2000) and to the lecture notes of Watrous (2011) for proper introductions to the field. This section provides a short glossary clarifying our notation and terminology in Section 2.1 followed by a review of two rarer but nonetheless simple and fundamental concepts from quantum information: the preservation of subsystem fidelity in Section 2.2 and the Bures angle in Section 2.3.

2.1. Terminology and notation.

Density matrix and quantum state. A *density matrix* or *quantum state* is a positive semidefinite matrix X with $\mathrm{Tr}(X) = 1$. Thus far, we have used upper-case Roman letters $(X, Y, \ldots)$ to denote density matrices, as well as other matrices. But it is standard practice in quantum information to denote density matrices with lower-case Greek letters $(\rho, \xi, \ldots)$. Hereafter, we adopt this convention.

Measurement operator. A *measurement operator* is a positive semidefinite matrix M with $\|M\| \leq 1$. Equivalently, it holds that $0 \preceq M \preceq I$.

Quantum channel. A *channel* is a completely positive and trace-preserving linear map $\Phi : \mathbb{M}_m \to \mathbb{M}_n$ from matrices to matrices. These maps correspond to physically realizable operations on quantum states.

Adjoint, matrix inner product. The *adjoint* A^* of a matrix A is simply the conjugate-transpose of A. The *inner product* $\langle A, B \rangle$ between two $m \times n$ matrices A, B is given by $\langle A, B \rangle = \text{Tr}(A^*B)$. The inner product between two k-tuples of matrices is given by the sum

$$\langle (A_1, \ldots, A_k), (B_1, \ldots, B_k) \rangle = \sum_{i=1}^{k} \langle A_i, B_i \rangle.$$

More generally, the adjoint Φ^* of a linear map Φ from matrices to matrices is the unique linear map with $\langle \Phi(X), Y \rangle = \langle X, \Phi^*(Y) \rangle$ for all X, Y. This formula extends in the obvious way to linear maps from tuples of matrices to tuples of matrices.

Trace norm. The *trace norm* $\|X\|_{\text{Tr}}$ of a matrix X is defined as the sum of the singular values of X. As a measure of distance between quantum states, the trace norm is given by

$$(2.1) \qquad \frac{1}{2}\|\rho - \xi\|_{\text{Tr}} = \max_{0 \preceq \Pi \preceq I} \langle \rho - \xi, \Pi \rangle$$

for all density matrices ρ, ξ.

Fidelity. The *fidelity* is another distance measure for quantum states given by

$$F(\rho, \xi) = \left\| \sqrt{\rho}\sqrt{\xi} \right\|_{\text{Tr}}$$

for all density matrices ρ, ξ.

2.2. Preservation of subsystem fidelity. Consider the following property of the fidelity function, which we call the *preservation of subsystem fidelity*: If ρ, ξ are states of a quantum system with fidelity $F(\rho, \xi)$ and ρ' is any state of a larger system consistent with

ρ, then it is always possible to find ξ' consistent with ξ such that $F(\rho',\xi') = F(\rho,\xi)$.

A formal construction of such a ξ' appears in Jain *et al.* (2009). Since their construction consists entirely of elementary matrix operations, there is an efficient parallel algorithm that takes as input ρ, ξ, ρ' and produces the desired state ξ' as output.

PROPOSITION 2.2 (Preservation of subsystem fidelity, (Jain *et al.* 2009)).   *Let* $\rho, \xi \in \mathbb{M}_m$ *and* $\rho' \in \mathbb{M}_{mn}$ *be density matrices with* $\mathrm{Tr}_{\mathbb{M}_n}(\rho') = \rho$. *There exists a density matrix* $\xi' \in \mathbb{M}_{mn}$ *with* $\mathrm{Tr}_{\mathbb{M}_m}(\xi') = \xi$ *and* $F(\rho',\xi') = F(\rho,\xi)$. *Moreover,* ξ' *can be computed efficiently in parallel given* ρ, ξ, ρ'.

2.3. The Bures angle.   The *Bures angle* or simply the *angle* $A(\rho,\xi)$ between quantum states ρ, ξ is defined by

$$A(\rho,\xi) \overset{\text{def}}{=} \arccos F(\rho,\xi).$$

The angle is a metric on quantum states, meaning that it is non-negative, equals zero only when $\rho = \xi$, and obeys the triangle inequality (Nielsen & Chuang 2000). Moreover, the angle is *contractive*, so that

$$A(\Phi(\rho), \Phi(\xi)) \le A(\rho,\xi)$$

for any quantum channel Φ. The Fuchs-van de Graaf inequalities establish a relationship between the fidelity and trace norm (Fuchs & van de Graaf 1999). The inequalities are

$$(2.3) \qquad 1 - F(\rho,\xi) \le \frac{1}{2}\|\rho - \xi\|_{\mathrm{Tr}} \le \sqrt{1 - F(\rho,\xi)^2}.$$

These inequalities can be used to derive a relationship between $A(\rho,\xi)$ and $\|\rho - \xi\|_{\mathrm{Tr}}$. For example,

PROPOSITION 2.4 (Relationship between trace norm and Bures angle).   *For all density matrices* ρ, ξ *it holds that*

$$\frac{1}{2}\|\rho - \xi\|_{\mathrm{Tr}} \le A(\rho,\xi) \le \sqrt{\frac{\pi}{2}\|\rho - \xi\|_{\mathrm{Tr}}}.$$

PROOF.   The lower bound on $A(\rho, \xi)$ follows immediately from Fuchs-van de Graaf:

$$\frac{1}{2}\|\rho - \xi\|_{\mathrm{Tr}} \leq \sqrt{1 - \cos A(\rho, \xi)^2} = \sin A(\rho, \xi) \leq A(\rho, \xi),$$

where we used the identity $\sin x \leq x$ for all $x \geq 0$.

To obtain the upper bound on $A(\rho, \xi)$, we employ the identity $\cos x \leq 1 - x^2/\pi$ for $x \in [0, \pi/2]$, which can be verified using basic calculus. Then, we have

$$\frac{1}{2}\|\rho - \xi\|_{\mathrm{Tr}} \geq 1 - \cos A(\rho, \xi) \geq \frac{A(\rho, \xi)^2}{\pi},$$

from which the proposition follows. $\square$

3. Rounding theorem for a relaxed min-max problem

In this section, we define a new min-max expression $\mu_\varepsilon(\mathbf{A}, \mathbf{P})$ that approximates the desired quantity $\lambda(\mathbf{A}, \mathbf{P})$ from (1.2) in the limit as ε approaches zero. This new expression is a relaxation of $\lambda(\mathbf{A}, \mathbf{P})$ that is more amenable to the MMW. We prove a "rounding theorem" (Theorem 3.1) by which near-optimal points for $\lambda(\mathbf{A}, \mathbf{P})$ are efficiently obtained from near-optimal points for $\mu_\varepsilon(\mathbf{A}, \mathbf{P})$. Our use of the Bures angle occurs in the proof of Lemma 3.5, which is used in the proof of our rounding theorem.

Define the relaxation $\mu_\varepsilon(\mathbf{A}, \mathbf{P})$ of $\lambda(\mathbf{A}, \mathbf{P})$ by

$$\mu_\varepsilon(\mathbf{A}, \mathbf{P})$$

$$\overset{\mathrm{def}}{=} \min_{(\rho_1, \ldots, \rho_k)} \max_{\substack{P \in \mathbf{P} \\ (\Pi_1, \ldots, \Pi_{k-1})}} \langle \rho_k, P \rangle + \frac{k}{\varepsilon} \sum_{i=1}^{k-1} \langle \mathrm{Tr}_{\mathbb{M}_n}(\rho_{i+1}) - \Phi_i(\rho_i), \Pi_i \rangle$$

$$= \min_{(\rho_1, \ldots, \rho_k)} \max_{P \in \mathbf{P}} \langle \rho_k, P \rangle + \frac{k}{\varepsilon} \sum_{i=1}^{k-1} \frac{1}{2} \|\mathrm{Tr}_{\mathbb{M}_n}(\rho_{i+1}) - \Phi_i(\rho_i)\|_{\mathrm{Tr}}.$$

Here, the minimum is taken over all density operators $\rho_1, \ldots, \rho_k \in \mathbb{M}_{mn}$ and the maximum over all $P \in \mathbf{P}$ and over all measurement operators $\Pi_1, \ldots, \Pi_{k-1} \in \mathbb{M}_m$. The second equality follows immediately from the identity (2.1) from Section 2.1.

Notice that the minimum in the definition of $\mu_\varepsilon(\mathbf{A}, \mathbf{P})$ is taken over *all* k-tuples $(\rho_1, \ldots, \rho_k)$ of density operators, not just those in $\mathbf{A}$. Each term in the summation serves to penalize any violation of the conditions required for membership in $\mathbf{A}$ by adding the magnitude of that violation to the objective function. The k/ε factor amplifies the penalty so as to remove incentive to select an element outside of $\mathbf{A}$. Indeed, it is clear that

$$\lim_{\varepsilon \to 0} \mu_\varepsilon(\mathbf{A}, \mathbf{P}) = \lambda(\mathbf{A}, \mathbf{P}).$$

The following "rounding" theorem establishes a specific rate of convergence for this limit. A subsequent extension of this theorem (Proposition 3.4) provides a means by which near-optimal points for $\lambda(\mathbf{A}, \mathbf{P})$ are efficiently computed from near-optimal points for $\mu_\varepsilon(\mathbf{A}, \mathbf{P})$.

THEOREM 3.1 (Rounding theorem). *For any $\varepsilon > 0$, it holds that* $\lambda(\mathbf{A}, \mathbf{P}) \geq \mu_\varepsilon(\mathbf{A}, \mathbf{P}) > \lambda(\mathbf{A}, \mathbf{P}) - \varepsilon$.

PROOF.   The first inequality is easy: Let $(\rho_1, \ldots, \rho_k)$ be optimal for $\lambda(\mathbf{A}, \mathbf{P})$, and let $(P, \Pi_1, \ldots, \Pi_{k-1})$ be optimal for $\mu_\varepsilon(\mathbf{A}, \mathbf{P})$. Then, we have

$$\lambda(\mathbf{A}, \mathbf{P}) \geq \langle \rho_k, P \rangle = \langle \rho_k, P \rangle + \frac{k}{\varepsilon} \sum_{i=1}^{k-1} \langle \mathrm{Tr}_{\mathbb{M}_n}(\rho_{i+1}) - \Phi_i(\rho_i), \Pi_i \rangle$$

$$\geq \mu_\varepsilon(\mathbf{A}, \mathbf{P}).$$

(The first inequality is because $(\rho_1, \ldots, \rho_k)$ is optimal for $\lambda(\mathbf{A}, \mathbf{P})$. The equality follows because $(\rho_1, \ldots, \rho_k) \in \mathbf{A}$, so each term in the sum is zero. The final inequality is because $(P, \Pi_1, \ldots, \Pi_{k-1})$ is optimal for $\mu_\varepsilon(\mathbf{A}, \mathbf{P})$.)

The second inequality is more difficult. We invoke the following lemma, the proof of which appears later in this section.

LEMMA 3.2 (Rounding lemma). *For any $\varepsilon > 0$ and any states* $\rho_1, \ldots, \rho_k \in \mathbb{M}_{mn}$, *there exists* $(\rho_1', \ldots, \rho_k') \in \mathbf{A}$ *such that*

$$\frac{1}{2}\|\rho_k - \rho_k'\|_{\mathrm{Tr}} < \varepsilon + \frac{k}{\varepsilon} \sum_{i=1}^{k-1} \frac{1}{2}\|\mathrm{Tr}_{\mathbb{M}_n}(\rho_{i+1}) - \Phi_i(\rho_i)\|_{\mathrm{Tr}}.$$

Moreover, $\rho'_1, \ldots, \rho'_k$ can be computed efficiently in parallel given $\rho_1, \ldots, \rho_k$.

Let $(\rho_1, \ldots, \rho_k)$ be optimal for $\mu_\varepsilon(\mathbf{A}, \mathbf{P})$, let $(\rho'_1, \ldots, \rho'_k)$ be the density operators obtained by invoking Lemma 3.2, and let $P \in \mathbf{P}$ be optimal for $\lambda(\mathbf{A}, \mathbf{P})$. Because $(\rho_1, \ldots, \rho_k)$ is optimal for $\mu_\varepsilon(\mathbf{A}, \mathbf{P})$, we have

$$(3.3) \qquad \mu_\varepsilon(\mathbf{A}, \mathbf{P}) \geq \langle \rho_k, P \rangle + \frac{k}{\varepsilon} \sum_{i=1}^{k-1} \frac{1}{2} \left\| \mathrm{Tr}_{\mathbb{M}_n}(\rho_{i+1}) - \Phi_i(\rho_i) \right\|_{\mathrm{Tr}}.$$

Employing the identity (2.1), the quantity $\langle \rho_k, P \rangle$ becomes

$$\langle \rho_k, P \rangle = \langle \rho'_k, P \rangle + \langle \rho_k - \rho'_k, P \rangle \geq \langle \rho'_k, P \rangle - \frac{1}{2} \left\| \rho_k - \rho'_k \right\|_{\mathrm{Tr}}.$$

Substituting the bound on $\frac{1}{2} \left\| \rho_k - \rho'_k \right\|_{\mathrm{Tr}}$ from Lemma 3.2, we see that the summation of trace norms in (3.3) is canceled, leaving

$$\mu_\varepsilon(\mathbf{A}, \mathbf{P}) > \langle \rho'_k, P \rangle - \varepsilon \geq \lambda(\mathbf{A}, \mathbf{P}) - \varepsilon$$

as desired. (The final inequality is because P is optimal for $\lambda(\mathbf{A}, \mathbf{P})$.) $\qquad \square$

PROPOSITION 3.4 (Construction of near-optimal strategies). *The following hold for any $\delta, \varepsilon > 0$:*

    (i) *If $(\rho_1, \ldots, \rho_k)$ is δ-optimal for $\mu_\varepsilon(\mathbf{A}, \mathbf{P})$, then there is an efficient parallel algorithm to compute $(\rho'_1, \ldots, \rho'_k) \in \mathbf{A}$ that is $(\delta + \varepsilon)$-optimal for $\lambda(\mathbf{A}, \mathbf{P})$.*

    (ii) *If $(P, \Pi_1, \ldots, \Pi_{k-1})$ is δ-optimal for $\mu_\varepsilon(\mathbf{A}, \mathbf{P})$, then P is also $(\delta + \varepsilon)$-optimal for $\lambda(\mathbf{A}, \mathbf{P})$.*

PROOF OF ITEM (I). Let $(\rho_1, \ldots, \rho_k)$ be δ-optimal for $\mu_\varepsilon(\mathbf{A}, \mathbf{P})$, let $(\rho'_1, \ldots, \rho'_k) \in \mathbf{A}$ be obtained by invoking Lemma 3.2, and let $P \in \mathbf{P}$. We have

$$\langle \rho'_k, P \rangle \leq \langle \rho_k, P \rangle + \frac{1}{2} \left\| \rho_k - \rho'_k \right\|_{\mathrm{Tr}}$$

$$\leq \langle \rho_k, P \rangle + \varepsilon + \frac{k}{\varepsilon} \sum_{i=1}^{k-1} \frac{1}{2} \left\| \mathrm{Tr}_{\mathbb{M}_n}(\rho_{i+1}) - \Phi_i(\rho_i) \right\|_{\mathrm{Tr}}$$

$$\leq \mu_\varepsilon(\mathbf{A}, \mathbf{P}) + \varepsilon + \delta \leq \lambda(\mathbf{A}, \mathbf{P}) + \varepsilon + \delta.$$

(The first inequality follows from (2.1); the second from Lemma 3.2; the third because $(\rho_1, \ldots, \rho_k)$ is δ-optimal for $\mu_\varepsilon(\mathbf{A}, \mathbf{P})$; and the fourth because $\mu_\varepsilon(\mathbf{A}, \mathbf{P}) \leq \lambda(\mathbf{A}, \mathbf{P})$.) It therefore follows that $(\rho_1', \ldots, \rho_k')$ is $(\delta + \varepsilon)$-optimal for $\lambda(\mathbf{A}, \mathbf{P})$. $\qquad\square$

PROOF OF ITEM (II). Let $(P, \Pi_1, \ldots, \Pi_{k-1})$ be δ-optimal for $\mu_\varepsilon(\mathbf{A}, \mathbf{P})$. For any $(\rho_1, \ldots, \rho_k) \in \mathbf{A}$, we have

$$\langle \rho_k, P \rangle = \langle \rho_k, P \rangle + \frac{k}{\varepsilon} \sum_{i=1}^{k-1} \langle \mathrm{Tr}_{\mathbb{M}_n}(\rho_{i+1}) - \Phi_i(\rho_i), \Pi_i \rangle$$

$$\geq \mu_\varepsilon(\mathbf{A}, \mathbf{P}) - \delta > \lambda(\mathbf{A}, \mathbf{P}) - \varepsilon - \delta.$$

(The equality is because $(\rho_1, \ldots, \rho_k) \in \mathbf{A}$, so each term in the sum is zero. The first inequality is because $(P, \Pi_1, \ldots, \Pi_{k-1})$ is δ-optimal for $\mu_\varepsilon(\mathbf{A}, \mathbf{P})$. The final inequality is because $\mu_\varepsilon(\mathbf{A}, \mathbf{P}) > \lambda(\mathbf{A}, \mathbf{P}) - \varepsilon$.) It therefore follows that P is $(\delta + \varepsilon)$-optimal for $\lambda(\mathbf{A}, \mathbf{P})$. $\qquad\square$

We now prove Lemma 3.2, the statement of which appeared in the proof of Theorem 3.1. Given any states $\rho_1, \ldots, \rho_k$, this lemma asserts that these states can be "rounded" to an element $(\rho_1', \ldots, \rho_k') \in \mathbf{A}$ in such a way that the distance between the final states ρ_k and ρ_k' is bounded by a function of the extent to which $(\rho_1, \ldots, \rho_k)$ violate the conditions required for membership in $\mathbf{A}$. Let us restate Lemma 3.2 in terms of the Bures angle.

LEMMA 3.5 (Rounding lemma). *For any $\varepsilon > 0$ and any states $\rho_1, \ldots, \rho_k \in \mathbb{M}_{mn}$, there exists $(\rho_1', \ldots, \rho_k') \in \mathbf{A}$ such that*

$$A(\rho_k, \rho_k') \leq \sum_{i=1}^{k-1} A\left(\mathrm{Tr}_{\mathbb{M}_n}(\rho_{i+1}), \Phi_i(\rho_i)\right).$$

Moreover, $\rho_1', \ldots, \rho_k'$ can be computed efficiently in parallel given $\rho_1, \ldots, \rho_k$.

PROOF. Define $\rho_1', \ldots, \rho_k'$ recursively as follows. Let $\rho_1' = \rho_1$. For each $i = 1, \ldots, k-1$ by the preservation of subsystem fidelity

([Proposition 2.2](#)), there exists ρ'_{i+1} (which can be computed efficiently in parallel) with $\text{Tr}_{\mathbb{M}_n}(\rho'_{i+1}) = \Phi_i(\rho'_i)$ and

$$A(\rho_{i+1}, \rho'_{i+1}) = A\left(\text{Tr}_{\mathbb{M}_n}(\rho_{i+1}), \Phi_i(\rho'_i)\right).$$

By the triangle inequality, this quantity is at most

$$A\left(\text{Tr}_{\mathbb{M}_n}(\rho_{i+1}), \Phi_i(\rho_i)\right) + A\left(\Phi_i(\rho_i), \Phi_i(\rho'_i)\right).$$

By contractivity of the Bures angle under channels, the summand on the right is at most $A(\rho_i, \rho'_i)$. The lemma now follows inductively from the fact that $A(\rho_1, \rho'_1) = 0$. □

It is easy to recover [Lemma 3.2](#) from [Lemma 3.5](#): It follows immediately from [Lemma 3.5](#) and [Proposition 2.4](#) (Relationship between trace norm and Bures angle) that

$$\frac{1}{2}\|\rho_k - \rho'_k\|_{\text{Tr}} \leq \sum_{i=1}^{k-1} \sqrt{\frac{\pi}{2} \left\|\text{Tr}_{\mathbb{M}_n}(\rho_{i+1}) - \Phi_i(\rho_i)\right\|_{\text{Tr}}}.$$

[Lemma 3.2](#) then follows from the fact that $\sqrt{\frac{\pi}{2}x} < \frac{1}{2\delta}x + \delta$ for all $x \geq 0$ and all $\delta > 0$.

4. A parallel oracle-algorithm for a min-max problem

In this section, we prove [Theorem 1.5](#) (Main result) by exhibiting an efficient parallel oracle-algorithm based on MMW for finding approximate solutions to the min-max problem [(1.2)](#). The precise formulation of the MMW method used in this paper is stated below as [Theorem 4.1](#). Our statement of this theorem is somewhat nonstandard: The result is usually presented in the form of an algorithm, whereas our presentation is purely mathematical. However, a cursory examination of the literature—say, [Kale (2007](#), Chapter 3)—reveals that our mathematical formulation is equivalent to the more conventional algorithmic form.

THEOREM 4.1 (Multiplicative weights update method, [Kale 2007](#)). *Fix $\gamma \in (0, 1/2)$ and $\alpha > 0$. Let $M^{(1)}, \ldots, M^{(T)}$ be arbitrary $d \times d$*

"loss" matrices with $0 \preceq M^{(t)} \preceq \alpha I$. Let $W^{(1)}, \ldots, W^{(T)}$ be $d \times d$ "weight" matrices given by

$$W^{(1)} = I \qquad W^{(t+1)} = \exp\left(-\gamma\left(M^{(1)} + \cdots + M^{(t)}\right)\right).$$

Let $\rho^{(1)}, \ldots, \rho^{(T)}$ be density operators obtained by normalizing each $W^{(1)}, \ldots, W^{(T)}$ so that $\rho^{(t)} = W^{(t)}/\operatorname{Tr}(W^{(t)})$. For all density operators ρ, it holds that

$$\frac{1}{T}\sum_{t=1}^{T}\langle\rho^{(t)}M^{(t)}\rangle \leq \langle\rho\frac{1}{T}\sum_{t=1}^{T}M^{(t)}\rangle + \alpha\left(\gamma + \frac{\ln d}{\gamma T}\right).$$

Note that Theorem 4.1 holds for *all* choices of loss matrices $M^{(1)}, \ldots, M^{(T)}$, including those for which each $M^{(t)}$ is chosen adversarially based upon $W^{(1)}, \ldots, W^{(t)}$. This adaptive selection of loss matrices is typical in implementations of the MMW.

Let us establish some notation before stating our algorithm. Let $\varepsilon > 0$ and consider the linear mapping $f_{\mathbf{A},\varepsilon}$ with the property that

$$\langle f_{\mathbf{A},\varepsilon}(\rho_1, \ldots, \rho_k), (P, \Pi_1, \ldots, \Pi_{k-1})\rangle$$
$$= \langle\rho_k, P\rangle + \frac{k}{\varepsilon}\sum_{i=1}^{k-1}\langle\operatorname{Tr}_{\mathbb{M}_n}(\rho_{i+1}) - \Phi_i(\rho_i), \Pi_i\rangle,$$

so that we may write

$$\mu_\varepsilon(\mathbf{A}, \mathbf{P}) = \min_{\substack{(\rho_1,\ldots,\rho_k)}} \max_{\substack{P \in \mathbf{P} \\ (\Pi_1,\ldots,\Pi_{k-1})}} \langle f_{\mathbf{A},\varepsilon}(\rho_1, \ldots, \rho_k), (P, \Pi_1, \ldots, \Pi_{k-1})\rangle.$$

It is clear that the mapping $f_{\mathbf{A},\varepsilon}$ is given by

$$f_{\mathbf{A},\varepsilon} : (\rho_1, \ldots, \rho_k)$$
$$\mapsto \left(\rho_k, \frac{k}{\varepsilon}\left[\operatorname{Tr}_{\mathbb{M}_n}(\rho_2) - \Phi_1(\rho_1)\right], \ldots, \frac{k}{\varepsilon}\left[\operatorname{Tr}_{\mathbb{M}_n}(\rho_k) - \Phi_{k-1}(\rho_{k-1})\right]\right).$$

It is tedious but straightforward to verify that the adjoint mapping $f_{\mathbf{A},\varepsilon}^*$ is given by

$$f_{\mathbf{A},\varepsilon}^* = \left(f_{\mathbf{A},\varepsilon,1}^*, \ldots, f_{\mathbf{A},\varepsilon,k}^*\right),$$

where

$$f^*_{\mathbf{A},\varepsilon,1} : (P, \Pi_1, \ldots, \Pi_{k-1}) \mapsto -\frac{k}{\varepsilon}\Phi^*_1(\Pi_1),$$

$$f^*_{\mathbf{A},\varepsilon,i} : (P, \Pi_1, \ldots, \Pi_{k-1}) \mapsto \frac{k}{\varepsilon}[\Pi_{i-1} \otimes I - \Phi^*_i(\Pi_i)]$$

$$\text{for } i = 2, \ldots, k-1,$$

$$f^*_{\mathbf{A},\varepsilon,k} : (P, \Pi_1, \ldots, \Pi_{k-1}) \mapsto P + \frac{k}{\varepsilon}\Pi_{k-1} \otimes I.$$

Note that for any $(P, \Pi_1, \ldots, \Pi_{k-1})$, it holds that

$$-\frac{k}{\varepsilon}I \preceq f^*_{\mathbf{A},\varepsilon,1}(P, \Pi_1, \ldots, \Pi_{k-1}) \preceq 0,$$

(4.2) $$-\frac{k}{\varepsilon}I \preceq f^*_{\mathbf{A},\varepsilon,i}(P, \Pi_1, \ldots, \Pi_{k-1}) \preceq \frac{k}{\varepsilon}I \qquad (2 \le i \le k-1),$$

$$0 \preceq f^*_{\mathbf{A},\varepsilon,k}(P, \Pi_1, \ldots, \Pi_{k-1}) \preceq \left(1 + \frac{k}{\varepsilon}\right)I \preceq \frac{2k}{\varepsilon}I.$$

Our MMW algorithm is stated as follows.

---

1. Let $\varepsilon = \delta/3$, let $\gamma = \frac{\varepsilon\delta}{12k^2}$, and let $T = \left\lceil \frac{\ln(mn)}{\gamma^2} \right\rceil$. Let $W_i^{(1)} = I \in \mathbb{M}_{mn}$ for each $i = 1, \ldots, k$.

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

    (a) For $i = 1, \ldots, k$: Compute the updated density operators $\rho_i^{(t)} = W_i^{(t)} / \operatorname{Tr}(W_i^{(t)})$.

    (b) For $i = 1, \ldots, k-1$: Compute the projection $\Pi_i^{(t)} \in \mathbb{M}_m$ onto the positive eigenspace of

    $$\operatorname{Tr}_{\mathbb{M}_n}(\rho_{i+1}^{(t)}) - \Phi_i(\rho_i^{(t)}).$$

    (c) Use the oracle to obtain a $\delta/3$-optimal solution $P^{(t)} \in \mathbb{M}_{mn}$ to the optimization problem for **P** (Problem 1.3) on input $\rho_k^{(t)}$.

    (d) Compute the loss matrices

    $$\left(M_1^{(t)}, \ldots, M_k^{(t)}\right)$$
    $$= \frac{\varepsilon}{2k^2}[f^*_{R,\varepsilon}\left(P^{(t)}, \Pi_1^{(t)}, \ldots, \Pi_{k-1}^{(t)}\right) + \frac{k}{\varepsilon}(I, \ldots, I, 0)].$$

(e) Update each weight matrix according to the standard MMW update rule:

$$W_i^{(t+1)} = \exp(-\gamma(M_i^{(1)} + \cdots + M_i^{(t)})).$$

3. Return

$$\tilde{\lambda} = \frac{1}{T} \sum_{t=1}^{T} \left\langle f_{R,\varepsilon}\left(\rho_1^{(t)}, \ldots, \rho_a^{(t)}\right), \left(P^{(t)}, \Pi_1^{(t)}, \ldots, \Pi_{k-1}^{(t)}\right)\right\rangle$$

as the δ-approximation to $\lambda(\mathbf{A}, \mathbf{P})$.

4. Compute

$$(\rho_1, \ldots, \rho_k) = \frac{1}{T} \sum_{t=1}^{T} (\rho_1^{(t)}, \ldots, \rho_k^{(t)})$$

$$(P, \Pi_1, \ldots, \Pi_{k-1}) = \frac{1}{T} \sum_{t=1}^{T} (P^{(t)}, \Pi_1^{(t)}, \ldots, \Pi_{k-1}^{(t)}),$$

the pair of which are $\frac{2}{3}\delta$-optimal for $\mu_\varepsilon(\mathbf{A}, \mathbf{P})$. Compute $(\rho_1', \ldots, \rho_k')$ from $(\rho_1, \ldots, \rho_k)$ as described in Proposition 3.4(i). Return $(\rho_1', \ldots, \rho_k')$ and P as the δ-optimal point for $\lambda(\mathbf{A}, \mathbf{P})$.

We are now ready to prove Theorem 1.5.

PROOF OF THEOREM 1.5.   We argue that the theorem is established by the above oracle-algorithm. To this end, note that each loss matrix $M_i^{(t)} \in \mathbb{M}_{mn}$ satisfies $0 \preceq M_i^{(t)} \preceq \frac{1}{k}I$—a fact that follows immediately from their definition in step 2d and the bounds (4.2) on the adjoint mapping $f_{\mathbf{A},\varepsilon}^*$.

For each $i = 1, \ldots, k$, it is clear that the construction of the density operators $\rho_i^{(t)}$ in terms of the loss matrices $M_i^{(t)}$ presented in the above oracle-algorithm is as defined in Theorem 4.1. It therefore follows that for any density operator $\rho_i^\star \in \mathbb{M}_{mn}$, we have

$$\frac{1}{T} \sum_{t=1}^{T} \langle \rho_i^{(t)} M_i^{(t)} \rangle \leq \langle \rho_i^\star \frac{1}{T} \sum_{t=1}^{T} M_i^{(t)} \rangle + \frac{1}{k}\left(\gamma + \frac{\ln(mn)}{\gamma T}\right).$$

Summing these inequalities over all i, we find that for any density operators $(\rho_1^\star, \ldots, \rho_k^\star)$, it holds that

$$\frac{1}{T} \sum_{t=1}^{T} \left\langle (\rho_1^{(t)}, \ldots, \rho_k^{(t)})(M_1^{(t)}, \ldots, M_k^{(t)}) \right\rangle$$

$$\leq \left\langle (\rho_1^\star, \ldots, \rho_k^\star) \frac{1}{T} \sum_{t=1}^{T} (M_1^{(t)}, \ldots, M_k^{(t)}) \right\rangle + \left(\gamma + \frac{\ln(mn)}{\gamma T} \right).$$

Substituting the definition of the loss matrices $M_i^{(t)}$ from step 2d and simplifying, we obtain

$$\tilde{\lambda} = \frac{1}{T} \sum_{t=1}^{T} \left\langle \left(\rho_1^{(t)}, \ldots, \rho_k^{(t)} \right), f_{R,\varepsilon}^* \left(P^{(t)}, \Pi_1^{(t)}, \ldots, \Pi_{k-1}^{(t)} \right) \right\rangle$$

$$\leq \left\langle (\rho_1^\star, \ldots, \rho_k^\star) \frac{1}{T} \sum_{t=1}^{T} f_{R,\varepsilon}^*(P^{(t)}, \Pi_1^{(t)}, \ldots, \Pi_{k-1}^{(t)}) \right\rangle$$

$$(4.3) \qquad + \underbrace{\frac{2k^2}{\varepsilon} \left(\gamma + \frac{\ln(mn)}{\gamma T} \right)}_{\text{error term}}.$$

Substituting the choice of γ, T from step 1, we see that the error term on the right side is at most $\delta/3$. Since this inequality holds for any choice of $(\rho_1^\star, \ldots, \rho_k^\star)$, it certainly holds for the optimal choice, from which it follows that the right side is at most $\mu_\varepsilon(\mathbf{A}, \mathbf{P}) + \delta/3$. By construction, each $(P^{(t)}, \Pi_1^{(t)}, \ldots, \Pi_{k-1}^{(t)})$ is a $\delta/3$-best response to $(\rho_1^{(t)}, \ldots, \rho_k^{(t)})$, so it must be that the left side of this inequality is at least $\mu_\varepsilon(\mathbf{A}, \mathbf{P}) - \delta/3$. It then follows from Theorem 3.1 (Rounding theorem) and the choice $\varepsilon = \delta/3$ that $|\tilde{\lambda} - \lambda(\mathbf{A}, \mathbf{P}))| < \frac{2}{3}\delta$ as desired.

Next, we argue that the point $(\rho_1', \ldots, \rho_k')$ returned in step 4 is δ-optimal for $\lambda(\mathbf{A}, \mathbf{P})$. By Proposition 3.4(i), it suffices to argue that $(\rho_1, \ldots, \rho_k)$ is $\frac{2}{3}\delta$-optimal for $\mu_\varepsilon(\mathbf{A}, \mathbf{P})$. To this end, choose any $(P^\star, \Pi_1^\star, \ldots, \Pi_a^\star)$. Since each $(P^{(t)}, \Pi_1^{(t)}, \ldots, \Pi_{k-1}^{(t)})$ is a $\delta/3$-best response to $(\rho_1^{(t)}, \ldots, \rho_k^{(t)})$, it holds that the inner product

$$\left\langle (\rho_1^{(t)}, \ldots, \rho_k^{(t)}) f_{R,\varepsilon}^*(P^{(t)}, \Pi_1^{(t)}, \ldots, \Pi_{k-1}^{(t)}) \right\rangle$$

can increase by no more than $\delta/3$ when $(P^\star, \Pi_1^\star, \ldots, \Pi_{k-1}^\star)$ is substituted for $(P^{(t)}, \Pi_1^{(t)}, \ldots, \Pi_{k-1}^{(t)})$. It then follows rom 6 that

$$\Big\langle \frac{1}{T} \sum_{t=1}^{T} (\rho_1^{(t)}, \ldots, \rho_k^{(t)}) f_{R,\varepsilon}^* \left(P^\star, \Pi_1^\star, \ldots, \Pi_{k-1}^\star \right) \Big\rangle$$
$$\leq \tilde{\lambda} + \delta/3 \leq \mu_\varepsilon(\mathbf{A}, \mathbf{P}) + \tfrac{2}{3}\delta,$$

and hence, $(\rho_1, \ldots, \rho_k)$ is $\tfrac{2}{3}\delta$-optimal for $\mu_\varepsilon(\mathbf{A}, \mathbf{P})$ as desired.

Next, we argue that the operator P returned in step 4 is δ-optimal for $\lambda(\mathbf{A}, \mathbf{P})$. By Proposition 3.4(ii), it suffices to argue that $(P, \Pi_1, \ldots, \Pi_{k-1})$ is $\tfrac{2}{3}\delta$-optimal for $\mu_\varepsilon(\mathbf{A}, \mathbf{P})$. To this end, choose any $(\rho_1^\star, \ldots, \rho_k^\star)$. It follows from 6 that

$$\big\langle (\rho_1^\star, \ldots, \rho_k^\star), f_{R,\varepsilon}^* \left(P, \Pi_1, \ldots, \Pi_{k-1} \right) \big\rangle \geq \tilde{\lambda} - \delta/3 \geq \mu_\varepsilon(\mathbf{A}, \mathbf{P}) - \tfrac{2}{3}\delta,$$

and hence, $(P, \Pi_1, \ldots, \Pi_{k-1})$ is $\tfrac{2}{3}\delta$-optimal for $\mu_\varepsilon(\mathbf{A}, \mathbf{P})$ as desired.

The efficiency of this algorithm is not difficult to argue. Each individual step consists only of matrix operations that are known to admit an efficient parallel implementation. Efficiency then follows from the observation that the number T of iterations is polynomial in k, $1/\delta$, and $\log(mn)$. $\square$

5. Double quantum interactive proofs

In this section, we prove DQIP $\subseteq$ PSPACE by means of Theorem 1.5. Specifically, in Section 5.2, we argue that the verifier in a double quantum interactive proof induces a min-max problem of the form (1.2) in which elements of $\mathbf{A}$ correspond to strategies for the yes-prover, elements of $\mathbf{P}$ correspond to strategies for the no-prover, and the value $\lambda(\mathbf{A}, \mathbf{P})$ corresponds to the probability with which the verifier rejects when both provers act optimally.

Thus, the parallel oracle-algorithm of Theorem 1.5—together with a parallel implementation of the oracle for optimization over $\mathbf{P}$—can be used to compute this probability to sufficient accuracy so as to determine which prover has the winning strategy. In Section 5.3, we provide a parallel implementation of the oracle required by Theorem 1.5. Finally, in Section 5.4, we recite the argument by which the existence of a parallel algorithm for approximating

$\lambda(\mathbf{A}, \mathbf{P})$ leads to the containment of DQIP inside PSPACE. First, we briefly introduce new notation in Section 5.1

5.1. Notation.   Until now, we have used the symbol $\mathbb{M}_n$ to denote the space of complex $n \times n$ matrices. This notation is ideal when only one or two distinct quantum systems are under consideration. However, discussion henceforth deals with many different systems (called *registers*), and so, we adopt the convention that distinct finite-dimensional complex vector spaces of the form $\mathbb{C}^d$ shall be denoted with calligraphic letters $(\mathcal{X}, \mathcal{Y}, \ldots)$. We also adopt the following notation:

$\mathcal{X}\mathcal{Y}$	Shorthand for the Kronecker product $\mathcal{X} \otimes \mathcal{Y}$. If $\mathcal{X} = \mathbb{C}^d$ and $\mathcal{Y} = \mathbb{C}^{d'}$ then $\mathcal{X}\mathcal{Y} = \mathbb{C}^{dd'}$.
$\mathbb{M}_{\mathcal{X}}$	The complex space of all linear operators (matrices) acting on $\mathcal{X}$.
$I_{\mathcal{X}} \in \mathbb{M}_{\mathcal{X}}$	The identity operator acting on $\mathcal{X}$.
$\mathrm{Tr}_{\mathcal{X}} : \mathbb{M}_{\mathcal{X}\mathcal{Y}} \to \mathbb{M}_{\mathcal{Y}}$	The partial trace over $\mathcal{X}$.

5.2. Characterization of strategies for the yes-prover.   The verifier in a double quantum interactive proof can be assumed to act upon two quantum registers: an m-qubit register M that is shared with the provers for the purpose of exchanging messages and a v-qubit register V that serves as a private memory for the verifier. Associated with the registers M, V are complex Euclidean spaces $\mathcal{M} = \mathbb{C}^{2^m}, \mathcal{V} = \mathbb{C}^{2^v}$, respectively. A verifier who exchanges a rounds of messages with the yes-prover followed by b rounds of messages with the no-prover is completely specified by a tuple $V = (|\psi\rangle, V_1, \ldots, V_{a+b-1}, \Pi)$, where

1. $|\psi\rangle \in \mathcal{M}\mathcal{V}$ is a pure state.

2. $V_1, \ldots, V_{a+b-1} \in \mathbb{M}_{\mathcal{M}\mathcal{V}}$ are unitary operators.

3. $\Pi \in \mathbb{M}_{\mathcal{M}\mathcal{V}}$ is a projective measurement operator.

The yes-prover acts upon the shared communication register M and a private memory register W with associated space $\mathcal{W}$. The actions of the yes-prover are specified by unitaries $A_1, \ldots, A_a \in \mathbb{M}_{\mathcal{M}\mathcal{W}}$.
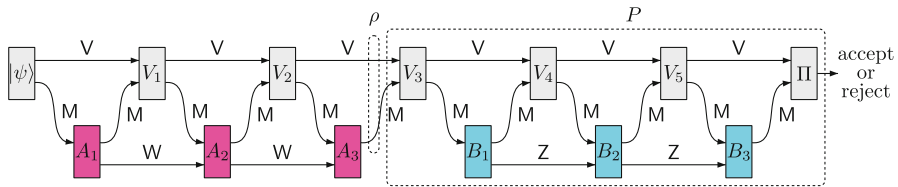
Figure 5.1:   An illustration of a double quantum interactive proof in which the verifier $V = (|\psi\rangle, V_1, \ldots, V_5, \Pi)$ exchanges $a = 3$ rounds of messages with the yes-prover followed by $b = 3$ rounds of messages with the no-prover before performing the measurement $\{\Pi, I - \Pi\}$ that dictates acceptance or rejection. Any choice of (A_1, A_2, A_3) and (B_1, B_2, B_3) induces a state ρ and a measurement operator P as indicated. The probability of rejection is given by $\langle \rho, P \rangle = \mathrm{Tr}(\rho P)$

Similarly, the no-prover acts upon the shared communication register M and a private memory register Z with associated space $\mathcal{Z}$. The actions of the no-prover are specified by unitaries $B_1, \ldots, B_b \in \mathbb{M}_{\mathcal{MZ}}$. The interaction proceeds as suggested by Figure 5.1 with measurement outcome Π indicating rejection.

Basic quantum formalism tells us that if the yes- and no-provers act according to $\vec{A} = (A_1, \ldots, A_a)$ and $\vec{B} = (B_1, \ldots, B_b)$, respectively, then the probability of rejection is given by

$$(5.1) \quad \begin{aligned} &\Pr\left[\text{reject} \mid \vec{A}, \vec{B}\right] \\ &= \|\Pi B_b V_{a+b-1} B_{b-1} \cdots B_1 V_a A_a V_{a-1} A_{a-1} \cdots A_2 V_1 A_1 |\psi\rangle\|^2. \end{aligned}$$

(For clarity, we have suppressed numerous tensors with identity and the initial states $|0\rangle$ of the provers' private memory registers.)

For any $\vec{A}$, let ρ be the reduced state of the verifier's registers (M, V) immediately after A_a is applied so that the actions of the yes-prover are completely represented by the state ρ. Similarly, for any $\vec{B}$, let P be the measurement operator on (M, V) obtained by bundling the verifier–no-prover interaction into a single measurement operator as suggested by Figure 5.1. The expression (5.1) for the probability of rejection can be rewritten in terms of ρ, P as

$$\Pr[\text{reject} \mid \vec{A}, \vec{B}] = \langle \rho, P \rangle.$$
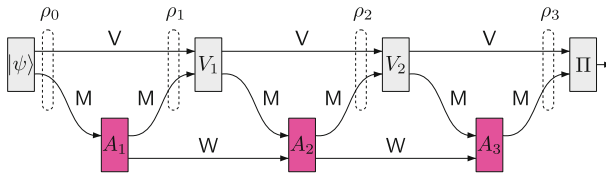
Figure 5.2: The states ρ_1, ρ_2, ρ_3 are a transcript of the referee's conversation with the yes-prover. It follows easily from the unitary equivalence of purifications that a triple (ρ_1, ρ_2, ρ_3) is a valid transcript if and only if it obeys the recursive relation $\text{Tr}_{\mathcal{M}_i}(\rho_i) = \text{Tr}_{\mathcal{A}_i}(V_{i-1}\rho_{i-1}V_{i-1}^*)$ for $i = 1, 2, 3$ where $V_0 = I$

By definition, the no-prover wishes to maximize this quantity, while the yes-prover wishes to minimize it. Let $\lambda(V)$ denote the verifier's probability of rejection when both provers act optimally. For a verifier with completeness c and soundness s, our goal is to determine whether $\lambda(V)$ is closer to $1 - c$ or to $1 - s$.

Let $\mathbf{Y}(V) \subset \mathbb{M}_{\mathcal{MV}}$ denote the set of states of (M, V) obtainable by the yes-prover, and let $\mathbf{P}(V) \subset \mathbb{M}_{\mathcal{MV}}$ denote the set of measurement operators on (M, V) obtainable by the no-prover. Then, the desired quantity $\lambda(V)$ is given by the min-max problem

$$(5.2) \qquad \lambda(V) = \min_{\rho \in \mathbf{Y}(V)} \max_{P \in \mathbf{P}(V)} \langle \rho, P \rangle.$$

What can be said of the sets $\mathbf{Y}(V), \mathbf{P}(V)$? Let us begin by considering the set $\mathbf{Y}(V)$. As suggested by Figure 5.2, each element of $\mathbf{Y}(V)$ can be viewed as the final entry ρ_a in a *transcript* $(\rho_1, \ldots, \rho_a)$ of the verifier's conversation with the yes-prover. Moreover, it is straightforward to use the unitary equivalence of purifications to characterize those a-tuples of density matrices which constitute valid transcripts. This characterization was first noted by Kitaev (2002).

PROPOSITION 5.3. *[Kitaev's consistency conditions, Kitaev 2002] Let $V = (|\psi\rangle, V_1, \ldots, V_{a+b-1}, \Pi)$ be a verifier, and let $\mathbf{Y}(V)$ be the set of admissible states for the yes-prover. A given state ρ is an element of $\mathbf{Y}(V)$ if and only if there exist density matrices*

$\rho_1, \ldots, \rho_a \in \mathbb{M}_{\mathcal{M}\mathcal{V}}$ with $\rho_a = \rho$ and

$$\mathrm{Tr}_{\mathcal{M}}(\rho_i) = \mathrm{Tr}_{\mathcal{M}}(V_{i-1}\rho_{i-1}V_{i-1}^*) \quad \text{for } i = 1, \ldots, a,$$

where we have written $V_0 = I$ and $\rho_0 = |\psi\rangle\langle\psi|$ for convenience.

With these observations in mind, we consider completely positive and trace-preserving linear maps

$$\Phi_0, \ldots, \Phi_{a-1} : \mathbb{M}_{\mathcal{M}\mathcal{V}} \to \mathbb{M}_{\mathcal{V}}$$

defined by

$$\Phi_0 : X \mapsto \mathrm{Tr}(X)\,\mathrm{Tr}_{\mathcal{M}}(|\psi\rangle\langle\psi|)$$
$$\Phi_i : X \mapsto \mathrm{Tr}_{\mathcal{M}}(V_i X V_i^*) \quad \text{for } i = 1, \ldots, a-1.$$

These maps specify the feasible region $\mathbf{A}(V)$ of an SDP of the form (1.1) from Section 1. Moreover, it follows from Kitaev's consistency conditions (Proposition 5.3) that $(\rho_0, \ldots, \rho_a) \in \mathbf{A}(V)$ if and only if $\rho_a \in \mathbf{Y}(V)$. Thus, the min-max problem (5.2) for $\lambda(V)$ can equivalently be written as

$$(5.4) \qquad \lambda(V) = \min_{(\rho_0,\ldots,\rho_a)\in\mathbf{A}(V)} \max_{P\in\mathbf{P}(V)} \langle \rho_a, P \rangle .$$

We have not yet shown that the set $\mathbf{P}(V)$ of measurement operators for the no-prover is compact and convex. But if we assume for the moment that it is, then we may already apply Theorem 1.5 so as to obtain a parallel oracle-algorithm for approximating $\lambda(V)$ on input $\Phi_0, \ldots, \Phi_{a-1}$ given an oracle for optimization over $\mathbf{P}(V)$.

5.3. Implementation of the oracle for best responses of the no-prover. In order to complete the description of our parallel algorithm for double quantum interactive proofs, it remains only to describe the implementation of the oracle for optimization for $\mathbf{P}(V)$ (Problem 1.3). In this section, we establish the following.

PROPOSITION 5.5. *Let $V = (|\psi\rangle, V_1, \ldots, V_{a+b-1}, \Pi)$ be a verifier, and let $\mathbf{P}(V)$ be the set of admissible measurement operators for the no-prover. There is a parallel algorithm for optimization over*

P(V) (*Problem 1.3*) *with run time bounded by a polynomial in* b, $1/\delta$, *and* $\log(\dim(\mathcal{MV}))$.

It follows that the algorithm of Section 4 yields an unconditionally efficient parallel algorithm for approximating $\lambda(V)$ given an explicit matrix representation of the verifier V.

As mentioned earlier, this instance of optimization over **P**(V) (Problem 1.3) will be rephrased as an SDP of the form (1.1) (plus some postprocessing) so that the algorithm of Section 4 can be reused in the implementation of our oracle.

To this end, choose any state $\rho \in \mathbb{M}_{\mathcal{MV}}$ and suppose that a (possibly cheating) yes-prover was somehow able to make it so that the registers (M, V) after the interaction with the yes-prover are in state ρ. Let W be a register large enough to admit a purification of ρ, and let $|\varphi\rangle \in \mathcal{WMV}$ be any such purification. If the no-prover acts according to $(B_1, \ldots, B_b)$, then the probability of rejection (as per Eq. (5.1)) is

$$\Pr[\text{reject} \mid \rho, (B_1, \ldots, B_b)] = \|\Pi B_b V_{a+b-1} B_{b-1} \cdots B_1 V_a |\varphi\rangle\|^2 \,.$$

Notice that this quantity also represents the probability of rejection in a different, single-prover interactive proof with a verifier V' whose initial state is $V_a|\varphi\rangle$. (Formally, the verifier V' exchanges b rounds of messages with one of the provers and zero messages with the other.) The unitaries $B_1, \ldots, B_b$ could specify actions for either the yes-prover or the no-prover—a choice that depends only upon how we label the components of the verifier V'.

Since our goal is to reduce optimization over **P**(V) (which is a maximization problem) to an SDP of the form (1.1) (which is a minimization problem), it befits us to view $B_1, \ldots, B_b$ as actions for the yes-prover in the interactive proof with verifier V'. Let us write

$$V' = (V_a|\varphi\rangle, V'_1, \ldots, V'_{b-1}, \Pi'),$$

where $V'_1, \ldots, V'_{b-1}, \Pi' \in \mathbb{M}_{\mathcal{MVW}}$ are given by

$$V'_i = V_{a+i} \otimes I_{\mathcal{W}} \qquad \text{for } i = 1, \ldots, b-1,$$
$$\Pi' = (I - \Pi) \otimes I_{\mathcal{W}}.$$

The private memory register V' of the new verifier V' is identified with the registers (V, W), and communication register M' of the new verifier is identified with M.

Each choice of unitaries $(B_1, \ldots, B_b)$ induces both a measurement operator $P \in \mathbf{P}(V)$ and a state $\xi \in \mathbf{Y}(V')$ with

$$\langle \rho, P \rangle = \| \Pi B_b V_{a+b-1} B_{b-1} \cdots B_1 V_a |\varphi\rangle \|^2 = 1 - \langle \xi, \Pi' \rangle,$$

and therefore,

$$\max_{P \in \mathbf{P}(V)} \langle \rho, P \rangle = 1 - \lambda(V') = 1 - \min_{\xi \in \mathbf{Y}(V')} \langle \xi, \Pi' \rangle.$$

Moreover, $P \in \mathbf{P}(V)$ achieves the maximum on the left side if and only if the unitaries $(B_1, \ldots, B_b)$ that induce P also induce a state $\xi \in \mathbf{Y}(V')$ that achieves the minimum on the right side.

Incidentally, by identifying elements of $\mathbf{P}(V)$ with elements of $\mathbf{A}(V')$, we have established that the set $\mathbf{P}(V)$ is compact and convex as required by Theorem 1.5. We are now ready to prove Proposition 5.5.

PROOF OF PROPOSITION 5.5.   Consider the following algorithm for optimization over $\mathbf{P}(V)$:

1. Use the algorithm of Section 4 to find $\xi \in \mathbf{Y}(V')$ minimizing $\langle \xi, \Pi' \rangle$.

2. Find the unitaries $(B_1, \ldots, B_b)$ that induce ξ. These unitaries also induce a measurement operator $P \in \mathbf{P}(V)$ maximizing $\langle \rho, P \rangle$. Compute P using $(B_1, \ldots, B_b)$ via standard matrix multiplication.

We already saw how the algorithm of Section 4 can be used to accomplish step 1 given an oracle for optimization over $\mathbf{P}(V')$. In this case, $\mathbf{P}(V') = \{\Pi'\}$ is a singleton set, and thus, the oracle for optimization over $\mathbf{P}(V')$ admits a trivial implementation by returning the only element.

It remains only to fill in the details for step 2. Recall that the algorithm of Section 4 finds a near-optimal transcript $(\xi_0, \ldots, \xi_b) \in$

A(V'), meaning that

$$\mathrm{Tr}_{\mathcal{M}}(\xi_1) = \mathrm{Tr}_{\mathcal{M}}(V_a|\varphi\rangle\langle\varphi|V_a^*),$$
$$\mathrm{Tr}_{\mathcal{M}}(\xi_{i+1}) = \mathrm{Tr}_{\mathcal{M}}(V_i'\xi_i V_i'^*) \qquad \text{for each } i = 1,\ldots, b-1.$$

(Here, ξ_0 is an arbitrary density matrix that is not used in our construction. The presence of this matrix is an artifact of the identification of **Y**(V') with **A**(V').) The following algorithm finds the unitaries $(B_1, \ldots, B_b)$:

1. Let $\mathcal{Z}$ be a space large enough to admit purifications of $\xi_1, \ldots, \xi_b$. Write $|\alpha_0\rangle = |\varphi\rangle|0_{\mathcal{Z}}\rangle$ and $V_0' = V_a$.

2. For each $i = 1, \ldots, b$:

    (a) Compute a purification $|\alpha_i\rangle \in \mathcal{ZMVW}$ of ξ_i.

    (b) Compute a unitary $B_i \in \mathbb{M}_{\mathcal{ZM}}$ that maps $V_{i-1}'|\alpha_{i-1}\rangle$ to $|\alpha_i\rangle$.

3. Return the desired unitaries $(B_1, \ldots, B_b)$.

Correctness of this construction is straightforward (though notationally cumbersome). Let us argue that each individual step consists only of matrix operations that are known to admit an efficient parallel implementation, from which it follows that the entire construction is efficient.

Step 2a requires that we compute a purification $|\alpha\rangle$ of a given mixed state ξ. This can be achieved by computing a spectral decomposition

$$\xi = \sum_i \mu_i |\phi_i\rangle\langle\phi_i|$$

of ξ; the purification $|\alpha\rangle$ is then given by

$$|\alpha\rangle = \sum_i \sqrt{\mu_i}|\phi_i\rangle|\phi_i\rangle.$$

Given two pure states $|\alpha\rangle, |\alpha'\rangle \in \mathcal{ZMVW}$ with

$$\mathrm{Tr}_{\mathcal{ZM}}(|\alpha\rangle\langle\alpha|) = \mathrm{Tr}_{\mathcal{ZM}}(|\alpha'\rangle\langle\alpha'|),$$

step 2b requires that we compute a unitary $B \in \mathbb{M}_{\mathcal{ZM}}$ that maps $|\alpha\rangle$ to $|\alpha'\rangle$. This can be achieved by computing Schmidt decompositions

$$|\alpha\rangle = \sum_i s_i |\phi_i\rangle |\psi_i\rangle, \qquad |\alpha'\rangle = \sum_i s'_i |\phi'_i\rangle |\psi_i\rangle$$

with respect to the partition $\mathcal{ZM} \otimes \mathcal{VW}$. (Schmidt decompositions on vectors are equivalent to singular value decompositions on matrices and hence can be implemented in parallel.) The desired unitary is then given by straightforward matrix multiplication and summation: $B = \sum_i |\phi'_i\rangle\langle\phi_i|$.   □

5.4. Containment of DQIP inside PSPACE.   The argument by which a parallel algorithm for double quantum interactive proofs leads to a proof of DQIP $\subseteq$ PSPACE is by now a familiar one. (See Section 3 of Jain *et al.* (2011) for a good exposition of this type of argument.)

PROOF OF THEOREM 1.6.   For each decision problem $L \in$ DQIP, we must prove that there is a polynomial-space algorithm for L. To this end, consider a "scaled up" version of NC known as NC(*poly*), which consists of all functions computable by polynomial-space uniform Boolean circuits of polynomial depth. It has long since been known that NC(*poly*) algorithms can be simulated in polynomial space (Borodin 1977), so in order to prove $L \in$ PSPACE, it suffices to give an NC(*poly*) algorithm for L.

Let V be a verifier with completeness c, soundness s, and polynomial-bounded p with $c - s \geq 1/p$ witnessing the membership of L in DQIP. Let x be any input string and consider the following algorithm for deciding whether x is a yes-instance or a no-instance of L:

1. Compute an explicit matrix representation of the verifier $V = (|\psi\rangle, V_1, \ldots, V_{a+b-1}, \Pi)$ on input x. As argued earlier, this representation specifies sets $\mathbf{A}(V), \mathbf{P}(V)$ for a min-max problem of the form (1.2).

2. Compute a δ-approximation of $\lambda(V)$ for the choice $\delta = (c - s)/3$ so as to determine which of the two provers has a winning strategy. Accept or reject accordingly.

The dimension $\dim(\mathcal{MV}) = 2^{m+v}$ of the matrix representation of a verifier on input x might grow exponentially in the bit length of x. Nevertheless, as argued in Jain *et al.* (2011) for ordinary quantum interactive proofs, it is not difficult to see that step 1 admits a straightforward implementation in NC(*poly*) via standard matrix multiplication.

Earlier in this section, we argued that the parallel oracle-algorithm of Theorem 1.5 can be used to compute the desired approximation of $\lambda(V)$. We also presented a parallel implementation of the oracle for optimization over $\mathbf{P}(V)$ required by Theorem 1.5. To see that this parallel algorithm is *efficient*, it suffices to observe that the number of rounds $a + b$ and the inverse of the accuracy parameter $1/\delta$ both scale as a polynomial in $|x|$ and hence also in $\log(\dim(\mathcal{MV}))$.

Thus, the above algorithm computes the composition of a function in NC(*poly*) with another function in NC. As NC(*poly*) is closed under such compositions, it follows that the above algorithm admits an NC(*poly*) implementation and hence also a polynomial-space implementation. It follows that $L \in$ PSPACE, and hence, DQIP $\subseteq$ PSPACE. $\qquad\square$

6. Consequences and extensions

6.1. A direct polynomial-space simulation of QIP. As mentioned in the introduction, a special case of our result is a direct polynomial-space simulation of multi-message quantum interactive proofs, resulting in a first-principles proof of QIP $\subseteq$ PSPACE. Recall that an ordinary, single-prover quantum interactive proof is a double quantum interactive proof in which the verifier exchanges zero messages with the no-prover. We already observed in Section 5.3 that such a verifier induces an SDP of the form (1.1) in which elements of the feasible region $\mathbf{A}$ are identified with strategies for the prover. In this case, Theorem 1.5 yields an efficient parallel algorithm for finding optimal strategies for the prover in a single-prover quantum interactive proof with no need to specify an oracle.

6.2. Finding near-optimal strategies.   The algorithm of Section 4 not only approximates the value $\lambda(\mathbf{A}, \mathbf{P})$ of the min-max problem (1.2), but it also finds near-optimal points $(\rho_1, \dots, \rho_k) \in \mathbf{A}$ and $P \in \mathbf{P}$. By contrast, in Section 5, we were primarily concerned with the problem of approximating only the value $\lambda(V)$ of the min-max problem (5.4). This quantity is the verifier's probability of rejection when both provers act optimally; approximating it suffices to prove DQIP $\subseteq$ PSPACE.

However, our result readily extends to the related search problem of *finding* near-optimal strategies for the provers. Indeed, step 4 of the algorithm of Section 4 returns a transcript $(\rho_0, \dots, \rho_a) \in \mathbf{A}(V)$ and a measurement operator $P \in \mathbf{P}(V)$, both of which are δ-optimal for $\lambda(V)$. The unitaries $(A_1, \dots, A_a)$ for the yes-prover can be recovered from the transcript $(\rho_0, \dots, \rho_a)$ via the method described in Section 5.3 with no additional complication.

It is only slightly more difficult to recover the no-prover's unitaries $(B_1, \dots, B_b)$ from P. Our definition of Problem 1.3 (optimization over $\mathbf{P}$) specifies only that a solution produce a near-optimal measurement operator $P \in \mathbf{P}$ for a given state ρ. But the algorithm for Problem 1.3 described in Section 5.3 for optimization over $\mathbf{P}(V)$ produces its output P by first constructing the associated unitaries $(B, \dots, B_b)$. It is a simple matter to modify our definition of Problem 1.3 so as to also return those unitaries in addition to P.

The near-optimal measurement operator P returned in step 4 of the algorithm of Section 4 is given by

$$P = \frac{1}{T} \sum_{t=1}^{T} P^{(t)},$$

which indicates a strategy for the no-prover that selects $t \in \{1, \dots, T\}$ uniformly at random and then acts according to $(B_1^{(t)}, \dots, B_b^{(t)})$. It is a simple matter to construct unitaries $(B_1, \dots, B_b)$ that implement this probabilistic strategy by sampling the integer t during the first round, recording that integer in the no-prover's private memory (which must be enlarged slightly to make room for it), and controlling the operation in subsequent turns on the contents of that integer. All of the matrix operations

required to construct $(B_1, \ldots, B_b)$ from each $(B_1^{(t)}, \ldots, B_b^{(t)})$ in this way can be implemented efficiently in parallel.

6.3. Robustness with respect to error. In Section 1.3.1, we noted that it is not immediately obvious that the classes DIP and DQIP are robust with respect to completeness and soundness parameters c, s. Because of this, we defined the classes to be inclusive as possible, allowing any verifier for which $c - s \geq 1/p$ for some polynomial-bounded function $p(|x|)$.

Nevertheless, it follows from the collapse of these classes to PSPACE that they are indeed robust with respect to completeness and soundness. In particular, classical interactive proofs for PSPACE (Lund *et al.* 1992; Shamir 1992) imply that if a decision problem L admits a double (quantum) interactive proof with $c - s \geq 1/p$, then L also admits a double (quantum) interactive proof with $c = 1$ and $s \leq 2^{-q}$ for any desired polynomial-bounded function $q(|x|)$.

However, the method by which the original verifier is transformed into the low-error verifier is very circuitous: The original verifier must be simulated in polynomial space according to Theorem 1.6 and then that polynomial-space computation must be converted back into an interactive proof with perfect completeness and exponentially small soundness according to proofs of IP = PSPACE. It would be nice to know whether a more straightforward transformation such as parallel repetition followed by a majority vote could be used to reduce error for double quantum interactive proofs and other bounded-turn interactive proofs with competing provers.

6.4. Arbitrary payoff observables. In the study of interactive proofs, attention is generally restricted to the *accept-reject* model, wherein the verifier's measurement $\{\Pi, I - \Pi\}$ indicates only acceptance or rejection without specifying a payout to the provers. From a game-theoretic perspective, one might wish to consider a more general verifier whose final measurement $\{\Pi_a\}_{a \in \Sigma}$ could have outcomes belonging to some arbitrary finite set Σ. In this case, the verifier awards *payouts* to the provers according to a *payout function* $v : \Sigma \to \mathbb{R}$ where $v(a)$ denotes the payout to the yes-

prover in the event of outcome a. (Since the game is zero-sum, the no-prover's payout must be $-v(a)$.)

Jain & Watrous (2009) describe a simple transformation by which their algorithm for one-turn quantum games can be used to approximate the expected payout in this more general setting. Their transformation extends without complication to double quantum interactive proofs.

In our case, the expected payout to the yes-prover when she and the no-prover play according to $(A_1, \ldots, A_a)$ and $(B_1, \ldots, B_b)$, respectively, is given by

$$\sum_{a \in \Sigma} v(a) \langle \phi | \Pi_a | \phi \rangle = \langle \phi | \Pi_\Sigma | \phi \rangle,$$

where

$$|\phi\rangle = B_b V_{a+b-1} B_{b-1} \cdots B_1 V_a A_a V_{a-1} A_{a-1} \cdots A_2 V_1 A_1 |\psi\rangle$$

is the final state of the system, and the Hermitian operator $\Pi_\Sigma = \sum_{a \in \Sigma} v(a) \Pi_a$ denotes the *payout observable* induced by the verifier. The expected payout of this interaction can be computed simply by translating and rescaling Π_Σ so as to obtain a measurement operator $0 \preceq \Pi \preceq I$ and then running our algorithm for double quantum interactive proofs with verifier $V = (|\psi\rangle, V_1, \ldots, V_{a+b-1}, \Pi)$. The expected payout of the original protocol is then obtained by inverting the scaling and translation operations by which Π was obtained from Π_Σ. As noted by Jain and Watrous, this transformation has the effect of inflating the additive approximation error δ by a factor of $\|\Pi_\Sigma\|$, which is the maximum absolute value of any given payout.

Acknowledgements

References

SANJEEV ARORA, ELAD HAZAN & SATYEN KALE (2005). The multiplicative weights update method: a meta algorithm and applications. Submitted.

ALLAN BORODIN (1977). On relating time and space to size and depth. *SIAM Journal on Computing* **6**(4), 733–744.

K. FAN (1953). Minimax Theorems. *Proceedings of the National Academy of Sciences* **39**, 42–47.

URIEL FEIGE & JOE KILIAN (1997). Making games short. In *Proceedings of the 29th ACM Symposium on Theory of Computing (STOC 1997)*, 506–516.

JOAN FEIGENBAUM, DAPHNE KOLLER & PETER SHOR (1995). A Game-Theoretic Classification of Interactive Complexity Classes. In *Proceedings of the 10th Conference on Structure in Complexity Theory*, 227–237.

LANCE FORTNOW, RUSSELL IMPAGLIAZZO, VALENTINE KABANETS & CHRISTOPHER UMANS (2008). On the complexity of succinct zero-sum games. *Computational Complexity* **17**(3), 353–376.

CHRISTOPHER FUCHS & JEROEN VAN DE GRAAF (1999). Cryptographic distinguishability measures for quantum mechanical states. *IEEE Transactions on Information Theory* **45**(4), 1216–1227. ArXiv:quant-ph/9712042v2.

Joachim von zur Gathen (1993). Parallel linear algebra. In *Synthesis of Parallel Algorithms*, John H. Reif, editor, chapter 13. Morgan Kaufmann Publishers, Inc.

Shafi Goldwasser & Michael Sipser (1989). Private coins versus public coins in interactive proof systems. In *Randomness and Computation*, Silvio Micali, editor, volume 5 of *Advances in Computing Research*, 73–90. JAI Press.

Gus Gutoski & John Watrous (2005). Quantum interactive proofs with competing provers. In *Proceedings of the 22nd Symposium on Theoretical Aspects of Computer Science (STACS'05)*, volume 3404 of *Lecture Notes in Computer Science*, 605–616. Springer. ArXiv:cs/0412102v1 [cs.CC].

Gus Gutoski & John Watrous (2007). Toward a general theory of quantum games. In *Proceedings of the 39th ACM Symposium on Theory of Computing (STOC 2007)*, 565–574. ArXiv:quant-ph/0611234v2.

Gus Gutoski & Xiaodi Wu (2012). Parallel approximation of min-max problems with applications to classical and quantum zero-sum games. In *Proceedings of the 27th IEEE Conference on Computational Complexity (CCC 2012)*, 21–31. ArXiv:1011.2787v2 [quant-ph].

Rahul Jain, Zhengfeng Ji, Sarvagya Upadhyay & John Watrous (2011). QIP=PSPACE. *Journal of the ACM* **58**(6), article 30.

Rahul Jain, Sarvagya Upadhyay & John Watrous (2009). Two-message quantum interactive proofs are in PSPACE. In *Proceedings of the 50th IEEE Symposium on Foundations of Computer Science (FOCS 2009)*, 534–543. ArXiv:0905.1300v1 [quant-ph].

Rahul Jain & John Watrous (2009). Parallel approximation of non-interactive zero-sum quantum games. In *Proceedings of the 24th IEEE Conference on Computational Complexity (CCC 2009)*, 243–253. ArXiv:0808.2775v1 [quant-ph].

Rahul Jain & Penghui Yao (2011). A parallel approximation algorithm for positive semidefinite programming. In *Proceedings of the 52nd IEEE Symposium on Foundations of Computer Science (FOCS 2011)*, 463–471. ArXiv:1104.2502v1 [cs.CC].

Rahul Jain & Penghui Yao (2012). A parallel approximation algorithm for mixed packing and covering semidefinite programs. ArXiv:1201.6090v1 [cs.DS].

Satyen Kale (2007). *Efficient algorithms using the multiplicative weights update method.* Ph.D. thesis, Princeton University.

Alexei Kitaev (2002). Quantum coin-flipping. Presentation at the 6th Workshop on *Quantum Information Processing* (QIP 2003).

Alexei Kitaev & John Watrous (2000). Parallelization, amplification, and exponential time simulation of quantum interactive proof system. In *Proceedings of the 32nd ACM Symposium on Theory of Computing*, 608–617.

Daphne Koller & Nimrod Megiddo (1992). The complexity of two-person zero-sum games in extensive form. *Games and Economic Behavior* **4**, 528–552.

Daphne Koller, Nimrod Megiddo & Bernhard von Stengel (1994). Fast algorithms for finding randomized strategies in game trees. In *Proceedings of the 26th ACM Symposium on Theory of Computing (STOC 1994)*, 750–759.

Michael Luby & Noam Nisan (1993). A Parallel Approximation Algorithm for Positive Linear Programming. In *Proceedings of the 25th ACM Symposium on Theory of Computing (STOC 1993)*, 448–457.

Carsten Lund, Lance Fortnow, Howard Karloff & Noam Nisan (1992). Algebraic methods for interactive proof systems. *Journal of the ACM* **39**(4), 859–868.

Chris Marriott & John Watrous (2005). Quantum Arthur-Merlin games. *Computational Complexity* **14**(2), 122–152. ArXiv:cs/0506068v1 [cs.CC].

Nimrod Megiddo (1992). A note on approximate linear programming. *Information Processing Letters* **42**(1), 53.

John von Neumann (1928). Zur Theorie der Gesellschaftspiele. *Mathematische Annalen* **100**(1), 295–320. In German.

Michael Nielsen & Issac Chuang (2000). *Quantum Computation and Quantum Information.* Cambridge University Press.

Christos Papadimitriou (1994). *Computational Complexity.* Addison-Wesley.

Richard Peng & Kanat Tangwongsan (2012). Faster and Simpler Width-Independent Parallel Algorithms for Positive Semidefinite Programming. ArXiv:1201.5135v1 [cs.DS].

Bill Rosgen & John Watrous (2005). On the hardness of distinguishing mixed-state quantum computations. In *Proceedings of the 20th Conference on Computational Complexity*, 344–354. ArXiv:cs/0407056v1 [cs.CC].

Maria Serna (1991). Approximating Linear Programming is Log-Space Complete for P. *Information Processing Letters* **37**(4), 233–236.

Adi Shamir (1992). IP = PSPACE. *Journal of the ACM* **39**(4), 869–877.

Luca Trevisan & Fatos Xhafa (1998). The Parallel Complexity of Positive Linear Programming. *Parallel Processing Letters* **8**(4), 527–533.

Manfred Warmuth & Dima Kuzmin (2006). Online variance minimization. In *Proceedings of the 19th Conference on Learning Theory*, volume 4505 of *Lecture Notes in Computer Science*, 514–528.

John Watrous (2011). Lecture Notes: Theory of Quantum Information. Available on the author's web page.

Xiaodi Wu (2010). Equilibrium Value Method for the Proof of QIP=PSPACE. ArXiv:1004.0264v2 [quant-ph].

Neal Young (2001). Sequential and parallel algorithms for mixed packing and covering. In *Proceedings of the 42nd IEEE Symposium on Foundations of Computer Science (FOCS 2001)*, 538–546.

Gus Gutoski
Institute for Quantum Computing,
School of Computer Science,
University of Waterloo,
Waterloo, ON, Canada.
gus.gutoski@uwaterloo.ca

Xiaodi Wu
Department of Electrical Engineering and Computer Science,
University of Michigan,
Ann Arbor, MI, USA.