

MIP^{co} = coRE

Junqiao (Randy) Lin

Centrum Wiskunde & Informatica

Qusoft

Amsterdam, Netherlands

Junqiao.Lin@cwi.nl

Abstract

In 2020, a landmark result by Ji, Natarajan, Vidick, Wright, and Yuen shows that MIP*, the class of languages that can be decided by a classical verifier interacting with multiple computationally unbounded provers sharing entanglement in the tensor product model, is equal to RE. We show that the class MIP^{co}, a complexity class defined similarly to MIP*, except with provers sharing the commuting operator model of entanglement, is equal to the class coRE. This shows that giving the provers two different models of entanglement leads to two completely different computational powers for interactive proof systems. Our proof builds upon the compression theorem used in the proof of MIP* = RE, and we use the tracially embeddable strategies framework to show that the same compression procedure in MIP* = RE also has the same desired property in the commuting operator setting. We also give a more streamlined proof of the compression theorem for non-local games by incorporating the synchronous framework used by Mousavi et al. [STOC 2022], as well as the improved Pauli basis test introduced by de la Salle [ArXiv:2204.07084].

We introduce a new equivalence condition for RE/coRE-complete problems, which we call the weakly compressible condition. We show that both MIP* and MIP^{co} satisfy this condition through the compression theorem, and thereby establish that the uncomputability for MIP* and MIP^{co} can be proved under a unified framework (despite these two complexity classes being different). Notably, this approach also gives an alternative proof of the MIP* = RE theorem, which does not rely on the preservation of the entanglement bound. In addition to non-local games, this new condition could also potentially be applicable to other promise problems.

CCS Concepts

• **Theory of computation** → **Quantum complexity theory**; **Interactive proof systems**; Turing machines.

Keywords

Non-local games, Interactive proof system, MIP*, MIP^{co}, RE-complete, coRE-complete, Compression

ACM Reference Format:

Junqiao (Randy) Lin. 2026. MIP^{co} = coRE. In *Proceedings of the 58th Annual ACM Symposium on Theory of Computing (STOC '26)*, June 22–26, 2026, Salt Lake City, UT, USA. ACM, New York, NY, USA, 9 pages. <https://doi.org/10.1145/3798129.3800751>



This work is licensed under a Creative Commons Attribution 4.0 International License. STOC '26, Salt Lake City, UT, USA

© 2026 Copyright held by the owner/author(s).

ACM ISBN 979-8-4007-2536-4/2026/06

<https://doi.org/10.1145/3798129.3800751>

1 Introduction

A two-prover non-local game $\mathcal{G}$ is played between a polynomial-time verifier and two computationally unbounded, but noncommunicating provers, which we name Alice and Bob. In this scenario, the verifier first samples a pair of questions (x, y) from a predetermined distribution μ and sends x to Alice (resp. y to Bob), who then responds with the answer a (resp. b). The verifier then computes a predicate function $D(x, y, a, b)$ that outputs either 0 or 1, with 1 indicating the verifier accepts (meaning the provers win the game), and 0 if the verifier rejects (meaning the provers lose the game). The provers know the initial question distribution and the predicate function and can strategize before the game, but cannot communicate during the game.

Non-local games are widely studied in the quantum information community. Famously, [7] shows that if the two provers employ the laws of quantum mechanics in their strategy, certain non-local games can be won with a higher probability, and this has since led to several experimental setups refuting the local realist model in our universe. Additionally, non-local games play an important role in quantum cryptography, both in the study of device-independent cryptography [8, 18], and in the analysis of certain quantum key exchange protocols [14].

When discussing models of entanglement in a non-local game, two natural models are considered. The first model, which is the more conventional model in the quantum information community, is the *tensor product model*. In this model, the provers are assumed to share an entangled state defined by a unit vector in a tensor factor of two potentially infinite-dimensional Hilbert spaces, $\mathcal{H}_A \otimes \mathcal{H}_B$. Alice, in this model, can make local measurements on the Hilbert space $\mathcal{H}_A$ and Bob similarly on the Hilbert space $\mathcal{H}_B$ in order to sample an answer pair (a, b) . The other, more general model is the *commuting operator model*, which is used in, e.g., the Haag-Kastler axioms of quantum mechanics [17]. This model defines the entangled state shared between the provers within a single Hilbert space $\mathcal{H}$. In this model, the provers are allowed to make measurements on the same Hilbert space as long as their measurement operators commute.

We call the optimal winning probability for a game $\mathcal{G}$ over all tensor product strategies the *tensor product value* of the game $\mathcal{G}$, or $\omega^*(\mathcal{G}) \in [0, 1]$, and the optimal winning probability over all commuting operator strategies as the *commuting operator value* of the game $\mathcal{G}$ or $\omega^{\text{co}}(\mathcal{G}) \in [0, 1]$. Since the commuting operator model includes the tensor product model (by considering the tensor product space as a single Hilbert space), we have $\omega^*(\mathcal{G}) \leq \omega^{\text{co}}(\mathcal{G})$ for all non-local games $\mathcal{G}$.

The complexity of non-local games. In computational complexity, non-local games are known as two-prover one-round *Multiprover Interactive Proof systems*, and these games are used to model the complexity class MIP.

When discussing a Multiprover Interactive Proof system, unless otherwise stated, we implicitly assume that it is the two-prover one-round variant. Roughly speaking, a language is in $\text{MIP} = \text{MIP}(\frac{2}{3}, \frac{1}{3})$ if every instance x of the language can be translated into a non-local game such that, if x is in the language, then the provers have a classical strategy that wins the game with a high probability ($\geq \frac{2}{3}$). If x is not in the language, then the provers cannot win the game with high probability ($\leq \frac{1}{3}$) given any classical strategy. Famously, [5] shows that $\text{MIP} = \text{NEXP}$, where NEXP is the set of languages decidable by a nondeterministic exponential-time Turing machine, and the technique used provided the foundation for showing the famous PCP theorem [2, 3].

The notion of a Multiprover Interactive proof system with “entangled provers” was first introduced in [9], where the computational model is defined similarly to MIP, except the provers are now given access to shared quantum entanglement. In this paper, we use MIP^* (resp. MIP^{co}) to denote the complexity class defined by multiprover interactive proof systems with the tensor product model of entanglement (resp. multiprover interactive proof systems with the commuting operator model of entanglement). For $t \in \{*, \text{co}\}$, the complexity class MIP^t is complete under polynomial time reduction with respect to the t -non-local game value problem, which is defined as a decision problem over the following two sets of non-local games:

$$\mathcal{L}_{\text{yes}}^t = \{\mathcal{G} : \omega^t(\mathcal{G}) = 1\} \quad \text{and} \quad \mathcal{L}_{\text{no}}^t = \left\{\mathcal{G} : \omega^t(\mathcal{G}) \leq \frac{1}{2}\right\}.$$

For clarity, we refer to the $*$ non-local game value problem as the *tensor product value problem* and the co non-local game value problem as the *commuting operator value problem*.

A recent breakthrough result by Ji et al. shows that $\text{MIP}^* = \text{RE}$ [19], where RE is the complexity class that contains all promise problems in which the “yes” case can be verified by a Turing machine in finite time. In other words, it is possible to reduce an instance of the halting problem to an instance of a non-local game $\mathcal{G}$ for which $\omega^*(\mathcal{G}) = 1$ if the Turing machine halts in a finite number of steps, and $\omega^*(\mathcal{G}) \leq \frac{1}{2}$ if the Turing machine does not halt. It is known that if the tensor product and the commuting operator value for a non-local game coincide, then one can construct a terminating algorithm that estimates the quantum value of the game up to some constant error [26]. The existence of such an algorithm, in conjunction with the $\text{MIP}^* = \text{RE}$ theorem, implies the existence of a game that has a commuting operator value strictly larger than its tensor product value. This, in turn, provides a negative answer to both Tsirelson’s problem in quantum information and Connes’s Embedding problem, a long-standing open problem in operator algebra [11, 28].

In contrast, another natural variant for MIP is MIP^{co} , which is defined similarly to MIP^* , but the provers are given access to the commuting operator model of entanglement instead. MIP^{co} is known to be in coRE by the algorithm known as the “NPA hierarchy” proposed in [26], and it has been conjectured to be coRE -complete [19, Section 1.4]¹.

As a main contribution of this paper, we give a positive answer to this conjecture.

THEOREM 1.1. $\text{MIP}^{\text{co}} = \text{coRE}$.

This provides a nice complementary result to $\text{MIP}^* = \text{RE}$, as it implies that employing two different axioms of quantum entanglement gives two completely different uncomputable verification powers to a MIP protocol. The proof of the main theorem follows by showing that the key technique used in $\text{MIP}^* = \text{RE}$, the gap compression theorem for non-local games, also holds in the commuting operator model. A key part of this adaptation relies on the recently discovered tracially embeddable strategies framework [22]. Then, we combine the gap compression theorem with the compression proof approach from [24, Section 1.1] to show that $\text{coRE} \leq_p \text{MIP}^{\text{co}}$. In conjunction with the NPA hierarchy algorithm, this also shows that all p -prover r -round MIP^{co} protocols are equivalent to the 2-prover 1-round MIP^{co} protocol.

We also streamline the proof for the gap compression theorem in this paper. Mainly, we incorporate some recent simplifications, such as the simplification to the Pauli basis test given in [12] and the synchronous game framework used in [24] for zero-gap MIP^* in our proof. Since we make use of the synchronous game framework, our result also implies that MIP_s^{co} , the complexity for the commuting operator value problem for synchronous games, is also coRE -complete.

The compressible condition. The proof of the $\text{MIP}^* = \text{RE}$ theorem relies on a key technique known as the gap compression theorem for non-local games. On a high level, the gap compression theorem shows the existence of an algorithm that takes a sequence of non-local games $\{\mathcal{G}_n\}_{n \in \mathbb{N}}$ with a polynomial time verifier and outputs a “compressed” sequence of non-local games $\{\mathcal{G}_n^{\text{Comp}}\}_{n \in \mathbb{N}}$ with a polylog time verifier. Furthermore, the compression procedure preserves the value for the tensor product value problem.

To be more precise, for all $n \in \mathbb{N}$, if $\omega^*(\mathcal{G}_n) = 1$, then $\omega^*(\mathcal{G}_n^{\text{Comp}}) = 1$ and if $\omega^*(\mathcal{G}_n) \leq \frac{1}{2}$, then $\omega^*(\mathcal{G}_n^{\text{Comp}}) \leq \frac{1}{2}$. The compression theorem is known as the “gap” compression theorem because it preserves the $\frac{1}{2}$ gap between the yes/no cases for the tensor product value problem. The gap compression given in [19] also has an additional clause on entanglement lower bound on the gap compression theorem, where the provers need a higher entanglement dimension, the dimension of the Hilbert space in which their joint entangled state is defined on, in the compressed game compared to the original game to formulate a strategy which wins with a probability of at least $\frac{1}{2}$. If we intuitively view the entanglement dimension as the amount of “resources” that the provers need for the game, then the compression theorem essentially states that it is possible for the verifier to perform “less work” when playing a non-local game with two entangled provers in the tensor product model. However, the provers would potentially have to prepare “more resources” in the form of an entangled state with a larger Hilbert space dimension in order to convince the verifier to accept the given game.

Based on the compression theorem, [19] constructs a game $\mathcal{G}$ for every Turing machine such that the provers need an entangled strategy whose entanglement dimensions (to succeed in the game with probability greater than $\frac{1}{2}$) grow with the runtime of the Turing machine. Hence, if the given Turing machine does not halt, then $\omega^*(\mathcal{G}) \leq \frac{1}{2}$, showing that $\text{RE} \leq_p \text{MIP}^*$. A similar observation was made in the first version of [27], where for certain

¹We remark that the co modifiers on both sides of $\text{MIP}^{\text{co}} = \text{coRE}$ refer to different things!

RE/coRE-complete problems such as the Halting problem and some versions of the word problem, a “resources-dependent” version of the compression theorem can be formulated.

Interestingly, [24, Theorem 6.10] shows that MIP* being RE-hard implies the existence of the gap compression theorem, which does not require the entanglement lower bound condition stated earlier. Since the entanglement lower bound condition is a specialized condition which only seems to apply in the context of non-local games with finite dimensional entanglement, an interesting question is whether this condition is necessary for showing that $\text{RE} \leq_p \text{MIP}^*$.

In this paper, we give a new condition for promise problems being RE/coRE-complete which we refer to as “compressible”. Intuitively, promise problems that are compressible admit a “compression theorem”, similarly to non-local games, but without the need for the preservation of resources like previous work. Using this new formulation, we give an alternative proof for $\text{RE} \leq_p \text{MIP}^*$, which only relies on the gap compression theorem and the existence of an algorithm that **halts in the “yes” case**, and **runs forever in the “no” case** for the tensor product value problem (this condition is trivially satisfied by $\text{MIP}^* \leq_p \text{RE}$). This, in turn, shows that the entanglement lower bound condition is **not** needed for the proof of $\text{MIP}^* = \text{RE}$. The same formulation can be used to show $\text{MIP}^{\text{co}} = \text{coRE}$, where a gap compression theorem in conjunction with the existence of an algorithm which **halts in the “no” case**, and **runs forever in the “yes” case** for the commuting operator value problem implies that $\text{coRE} \leq_p \text{MIP}^{\text{co}}$. Since our formulation works for general promise problems, we believe our approach can be generalized to establish the conjectured RE/coRE-completeness for other promise problems.

Explicit separation between the models of entanglement. As a corollary of the $\text{MIP}^* = \text{RE}$ theorem, the tensor product model of entanglement is mathematically different from the commuting operator model of entanglement. A natural follow-up question is whether we can find an experimental setup similar to the Bell test scenario to determine which is the right way to model entanglement in our universe? [19, Theorem 12.10] shows the existence of a game $\mathcal{G}$ such that $\omega^*(\mathcal{G}) \leq \frac{1}{2}$ and $\omega^{\text{co}}(\mathcal{G}) = 1$, which, in theory, could serve as the Bell test mentioned earlier. However, the question and answer sets for the given game have a magnitude of 10^{20} , making it impractical for experimental implementation.

We show that given a non-local game $\mathcal{G}$, the promise problem of deciding whether $\omega^*(\mathcal{G}) = \omega^{\text{co}}(\mathcal{G})$ or $|\omega^*(\mathcal{G}) - \omega^{\text{co}}(\mathcal{G})| \geq c$ for any fixed constant $c \in [0, 1]$ is RE-complete. In other words, there is no algorithm which can be used to find explicit separation between the tensor product model and the commuting operator model for general non-local games! On the brighter side, our proof also gives a reduction for any non-halting Turing machine to an instance of a game such that $|\omega^*(\mathcal{G}) - \omega^{\text{co}}(\mathcal{G})| \geq c$; however, since the technique used is similar to the one given in [19, Theorem 12.10], we suspect that the question size and answer size would be as large as those in [19].

Consequences. An obvious consequence of the main result of this paper is that **there is no algorithm which can estimate the commuting operator value of the game** up to any constant $c \in (0, 1)$, or else one can use this algorithm to construct a halting algorithm for the co-non-local game value problem. Our result also

implies that one cannot compute the convergence rate of the NPA hierarchy in general.

In conjunction with recent work in operator algebra, our main result also gives a negative result to a stronger variant of Connes embedding conjecture and Kirchberg’s QWEP conjecture. Roughly speaking, the disproof of the Connes embedding problem states that every tracial von Neumann algebra on a separable Hilbert space cannot be approximated by a limit of finite-dimensional matrices. A recent result by [4] shows that our main result also implies that furthermore all tracial von Neumann algebras cannot be approximated by *any* computable object, thus showing the class of tracial von Neumann algebras is “uncomputable” in nature.

The disproof of Kirchberg’s QWEP conjecture states that the maximum tensor product (or the “algebraic” tensor product) norm for $C^*(\mathcal{F}_n \times \mathcal{F}_n)$ (where $\mathcal{F}_n$ denotes the free group with n generators) is different than the minimum tensor product (or the “analytic” tensor product) norm. By using our main result, [16] shows that in general, there is no algorithm which can approximate the maximum tensor product norm of $C^*(\mathcal{F}_n \times \mathcal{F}_n)$ for all $n \in \{2, 3, \dots\}$. By taking the case where $n = 2$, this also gives a negative answer to [15, Problem 4.2]².

2 Compressible Condition

To demonstrate the idea behind the compressible condition, we begin by presenting a simple example with decision problems. We assume the standard Turing machine as the model of computation in this paper. For a Turing machine T , we use $|T|$ to denote the description length of the Turing machine. To abuse notation slightly, we also use the same notation to denote both the description of the Turing machine and the function that the Turing machine implements. We also take the convention that every integer given as an input for a Turing machine is being represented under its binary representation. This means that any integer n will be treated as an $O(\text{polylog}(n))$ -bit input for all Turing machines. We give the definition of a compressible language below.

Definition 2.1 (Compressible promise problem). Let $L \subseteq \{0, 1\}^*$. We say that L is *compressible* if there exists a universal algorithm Compress^L with the following properties:

- (1) (Output) Compress^L , which takes as input a description of a Turing machine Seq_L , and outputs a description of a Turing machine $\text{Seq}_L^{\text{Comp}}$, computes the function $\text{Seq}_L^{\text{Comp}} : \mathbb{N} \rightarrow \{0, 1\}^*$ in $O(\text{polylog}(n))$ time.
- (2) (Runtime): Compress^L runs in $O(|\text{Seq}_L|)$ time.
- (3) If Seq_L implements a function which maps $\mathbb{N} \rightarrow \{0, 1\}^*$ and runs in $O(\text{poly}(n))$ time, then for all $n \in \mathbb{N}$, the following holds:
 - (Completeness): If $\text{Seq}_L(n) \in L$, then $\text{Seq}_L^{\text{Comp}}(n) \in L$,
 - (Soundness): If $\text{Seq}_L(n) \notin L$, then $\text{Seq}_L^{\text{Comp}}(n) \notin L$.

We point out that based on the above definition, if L is compressible, then the language $\text{co}L = \{0, 1\}^* \setminus L$ is also compressible. An important remark about the compressible condition is that the Compress^L algorithm is language dependent. In other words, Compress^L takes **any** Turing machine which computes a sequence of strings in $O(\text{poly}(n))$ time and maps it to a Turing machine that

²Thus showing that “no, you can not compute the operator norm”!

computes a sequence of strings with a significantly smaller runtime while still preserving whether the n th string of the output is in L or not in L .

We remark that this is an unnatural condition for a language L , as $\text{Seq}_L^{\text{Comp}}$, when generating the n th instance, cannot generate the n th instance of Seq_L due to the smaller runtime requirement and, hence, cannot even decide whether $\text{Seq}_L(n) \in L$ or $\text{Seq}_L(n) \notin L$. The Compress^L algorithm has to, in some way, manipulate the description of the Turing machine Seq_L in a black-box way to reduce the runtime. We use the following clever example given in [27] to show that the halting problem is compressible.

Example 2.2 (The Halting problem is compressible). Recall, the halting problem is defined as $L = \{\langle \text{Prog} \rangle\}$, where $\langle \text{Prog} \rangle$ is a binary encoding of a Turing machine which halts on the empty input. Define compression algorithm $\text{Compress}^{\text{HALT}}$ for the halting problem as follows: given the input $\langle \text{Seq}_{\text{HALT}} \rangle$, $\text{Compress}^{\text{HALT}}$ returns a description of $\langle \text{Seq}_{\text{HALT}}^{\text{Comp}} \rangle$, described by Pseudocode 1.

- 1 **Input:** Integer n
- 2 Compute and return the description of $\langle \text{Prog}_n^{\langle \text{Seq}_{\text{HALT}} \rangle} \rangle$, where the pseudocode for $\text{Prog}_n^{\langle \text{Seq}_{\text{HALT}} \rangle}$ is given in Pseudocode 2

Pseudocode 1: The description of $\text{Seq}_{\text{HALT}}^{\text{Comp}}$.

- 1 Compute the description of $\langle \text{Seq}_{\text{HALT}}(n) \rangle$ using $\langle \text{Seq}_{\text{HALT}} \rangle$ and n , halt and return ERROR if $\langle \text{Seq}_{\text{HALT}} \rangle$ is not a valid description for a Turing machine.
- 2 Run the program $\langle \text{Seq}_{\text{HALT}}(n) \rangle$, halt and return ERROR if $\langle \text{Seq}_{\text{HALT}}(n) \rangle$ is not a valid description for a Turing machine.

Pseudocode 2: The description of the output for $\text{Prog}_n^{\langle \text{Seq}_{\text{HALT}} \rangle}$. We remark that the above program depends on both Seq_{HALT} and n

In the above example, one should interpret the description of $\langle \text{Prog}_n^{\langle \text{Seq}_{\text{HALT}} \rangle} \rangle$ given in Pseudocode 2 as an instance for the halting problem (and hence, the runtime of line 1 from Pseudocode 2 is irrelevant). The program $\text{Seq}_{\text{HALT}}^{\text{Comp}}$ merely generates different instances of the halting problem by outputting the description of $\langle \text{Prog}_n^{\langle \text{Seq}_{\text{HALT}} \rangle} \rangle$.

Since the Turing machine $\text{Seq}_{\text{HALT}}^{\text{Comp}}(n)$ only returns the description for $\langle \text{Prog}_n^{\langle \text{Seq}_{\text{HALT}} \rangle} \rangle$ which depends on n , the description of $\langle \text{Seq}_{\text{HALT}}^{\text{Comp}}(n) \rangle$ can be computed in $O(\text{poly}(|\langle \text{Seq} \rangle|))$ time. Since any integer input is represented under the binary representation, $\text{Seq}_D^{\text{Comp}}$ runs in $O(\text{polylog}(n))$. Furthermore, for all integer $n \in \mathbb{N}$

- If $\text{Seq}_{\text{HALT}}(n)$ returns a description of a halting program, then $\text{Seq}_{\text{HALT}}^{\text{Comp}}(n) = \langle \text{Prog}_n^{\langle \text{Seq}_{\text{HALT}} \rangle} \rangle$ is a description of a halting program.
- Otherwise, if $\text{Seq}_{\text{HALT}}(n)$ returns a description of a non-halting program, then $\text{Seq}_{\text{HALT}}^{\text{Comp}}(n) = \langle \text{Prog}_n^{\langle \text{Seq}_{\text{HALT}} \rangle} \rangle$ is a description of a non-halting program.

This shows that the halting problem is an example of a compressible promise problem.

Notably, the above example shows that all decision problems in RE are compressible. Given a compressible language L that is also “non-trivially” in RE, our goal is to show that L must be RE-complete. We first give a more precise definition for L being “non-trivially” in RE by making the following assumption:

- (1) $L \in \text{RE}$. In other words, there exists a Turing machine $\text{Algo}_L : \{0, 1\}^* \rightarrow \{0, 1\}$, such that Algo_L , when running on the input $x \in L$, halts in finite time and outputs 1, and **runs forever** if given input $y \notin L$ (this can be achieved by changing the termination condition for the “no” case for Algo_L to running an infinite loop).
- (2) L is not trivial, meaning $|L| = |(\{0, 1\}^* \setminus L)| = \infty$. There exist $x_{\text{yes}} \in L$ and $x_{\text{no}} \in \{0, 1\}^* \setminus L$ which are both trivially computable.
- (3) We can generate an instance $x_{\text{yes}} \in L$, and an instance $x_{\text{no}} \notin L$.

We now show that $\text{RE} \leq L$ (where $L_1 \leq L_2$ implies that there exists a mapping reduction from L_1 to L_2). Let $\mathcal{E}$ be a Turing machine. We wish to reduce $\mathcal{E}$ into an instance $x_{\mathcal{E}} \in \{0, 1\}^*$ such that $|x_{\mathcal{E}}| = \text{poly}(|\mathcal{E}|)$ and

- If $\mathcal{E}$ halts in a finite number of steps, then $x_{\mathcal{E}} \in L$,
- If $\mathcal{E}$ does not halt, then $x_{\mathcal{E}} \notin L$.

- 1 **Input:** Integer n .
- 2 Run $\mathcal{E}$ for n steps. If $\mathcal{E}$ halts in the given steps, **return** x_{yes} , the yes-instance guaranteed by the non-triviality condition above.
- 3 Compute $\langle \text{Seq}_L \rangle$, the description for Seq_L .
- 4 Simulate the following program describe by step 6 and 7 for n steps
- 5 Compute $\text{Seq}_L(1)$
- 6 Run Algo_L , the RE algorithm for L guaranteed by assumption 1 above, on the computed instance $\text{Seq}_L(1)$.
- 7 If the above subroutine halts in n steps, **return** x_{no} , the no-instance guaranteed by the non-triviality condition.
- 8 Compute $\text{Compress}^L(\langle \text{Seq}_L \rangle)$ and obtain the description for $\langle \text{Seq}_L^{\text{Comp}} \rangle$.
- 9 **Return** $\text{Seq}_L^{\text{Comp}}(n + 1)$.

Pseudocode 3: The description of Seq_L for demonstrating the generalized compression framework.

Consider the following function $\text{Seq}_L : \mathbb{N} \rightarrow \{0, 1\}^*$ given by Pseudocode 3. We remark that line 3 uses a self-referential trick to make Seq_L perform computation steps on its own source code. This step can be done in polynomial time with respect to the description length of Seq_L using Kleene’s recursion theorem, and we refer to [21, Chapter 14.2] and [29, Chapter 6.1] for more details.

We first analyze the runtime of Seq_L . Lines 2, 4 and 7 of Pseudocode 3 trivially take time $O(n)$. By the recursion theorem mentioned earlier, we see that lines 3 and 8 take time based on the

description length of $\langle \text{Seq}_L \rangle$, independent of n . By looking at Pseudocode 3, we see that the description length of $\langle \text{Seq}_L \rangle$ depends **only** on the description length of $\mathcal{E}$. Line 9 takes time $O(\text{polylog}(n))$ by the definition of Compress^L . Hence, the runtime for $\text{Seq}_L(n)$ is $O(n \cdot \log(n) + \text{poly}(|\mathcal{E}|)) = O(\text{poly}(n))$ (since $\mathcal{E}$ is a fixed Turing machine, its description length is independent of the variable n). Thus, by the definition of Compress^L , since Seq_L runs in $O(\text{poly}(n))$ time, the compression step on line 9 outputs a Turing machine that is a “compressed” version of Seq_L .

We claim that $x_{\mathcal{E}} = \text{Seq}_L(1)$ is the desired instance for the reduction. We first argue that Seq_L never halts on line 7 of Pseudocode 3 given **any** input $n \in \mathbb{N}$. Suppose, for a contradiction, that there exists some $C \in \mathbb{N}$ such that the algorithm Seq_L halts in line 7 of Pseudocode 3 with C as the input; We can also assume that C is the smallest integer such that this holds without loss of generality. The goal is to argue that whenever $\text{Seq}_L(C)$ does halt in line 7, then $\text{Seq}_L(1)$ simultaneously belongs in L and does not belong in L , thus creating a contradiction.

By first analysing Pseudocode 3, we see that Seq_L halting in step 2 implies that it cannot halt in step 7. This also means that $\mathcal{E}$ cannot halt in C steps, and hence Seq_L cannot halt in step 2 of Pseudocode 3 for all input $n < C$.

Now, Seq_L halting in line 7 on input C implies that $\text{Seq}_L(C) = x_{\text{no}} \notin L$. Seq_L halting in line 7 also implies that $\text{Algo}_L(\text{Seq}_L(1))$ halts in a finite number of steps, which means $\text{Seq}_L(1) \in L$ by the definition of Algo_L . By the minimal assumption of C and the argument above, Seq_L does not terminate on line 2 or 7 for all inputs $1 \leq n < C$, and hence $\text{Seq}_L(C-1) = \text{Seq}_L^{\text{comp}}(C)$. By the compressible condition, since $\text{Seq}_L^{\text{comp}}(C) \notin L$, this implies that $\text{Seq}_L^{\text{comp}}(C) \notin L$ and hence $\text{Seq}_L(C-1) \notin L$. By a simple inductive argument, one can deduce that $\text{Seq}_L(1) \notin L$. This creates the contradiction needed to argue that Pseudocode 3 never terminates via the exit clause on line 7. Notably, this also implies that Algo_L , with $\text{Seq}_L(1)$ as the input, **cannot** halt before $\mathcal{E}$ (or else Seq_L would halt in the exit clause on line 7, thus creating the given above contradiction).

Now, suppose $\mathcal{E}$ halts in C steps, by construction, we have $\text{Seq}_L(C) \in L$, and hence, by a similar inductive argument as above, we see that $\text{Seq}_L(1) \in L$. If $\mathcal{E}$ does not halt, then by the above argument, we see that Algo_L also cannot halt given the input $\text{Seq}_L(1)$. This means that Algo_L does not halt when given $\text{Seq}_L(1)$ as the input, implying that $\text{Seq}_L(1) \notin L$. This shows that L is RE-complete. We remark that this reduction is poly-time, as $x_{\mathcal{E}} = \text{Seq}_L(1)$ which can be computed in $O(\text{poly}(|\mathcal{E}|))$ time.

If a language L is shown to be compressible and in coRE, $\{0, 1\}^* \setminus L$ is compressible and in RE, and hence L is coRE. Intuitively, the above argument relies on the fact that we can embed the RE algorithm into a uniform Turing machine which generates a sequence of promise problems for the given language and use compression to “infinitely” reduce the runtime of the algorithm. The above proof approach for showing $L \leq \text{RE}$ is a generalization of the conjectured approach for showing $\text{coRE} \subseteq \text{MIP}^{\text{co}}$ in [24, Pseudocode 4]. In comparison to the first draft of [27], there is no dependency on some “resource” in the Compress^L map. Interestingly, as pointed out from [27], the Halting Problem is also compressible, which means that any RE-complete

language is also compressible, and we present their formulation in Theorem 2.2 of this paper.

Weakly compressible condition. In order to apply the compressible condition to non-local games, we have to first reformulate the compressible condition to hold for promise problems. For a decision problem D given by $D_{\text{yes}} \subseteq \{0, 1\}^*$ and $D_{\text{no}} \subseteq \{0, 1\}^*$, we define D to be compressible if it admits a similar Compress^D algorithm which compresses a *uniform problem instance* for D , or a Turing machine $\text{Seq}_D : \mathbb{N} \rightarrow D_{\text{yes}} \cup D_{\text{no}}$. To draw the parallel to the definition given in Theorem 2.1, one can interpret $\text{Seq}_L : \mathbb{N} \rightarrow \{0, 1\}$ given in the previous section as a function which maps $n \in \mathbb{N}$ to an element to either in L or $\{0, 1\}^* \setminus L$. In this case, the Compress^D generates a description of a “compressed” uniform problem instance $\text{Seq}_D^{\text{comp}}$ which has the same completeness/soundness property given in Theorem 2.1 (i.e. for $i \in \{\text{yes}, \text{no}\}$ If $\text{Seq}_D(n) \in D_i$, then $\text{Seq}_D^{\text{comp}}(n) \in D_i$ for all $n \in \mathbb{N}$), assuming the given input is a uniform problem instance for D which runs in $O(\text{poly}(n))$.

Unfortunately, this generalization in practice is very hard to show for any non-trivial language. One of the reasons is that the compressible condition requires one to show the existence of a universal compressible map which works for all $O(\text{poly}(n))$ uniform problem instances. In an effort to make this condition more applicable for general promise problems, we define a weaker notion of the compressible condition known as weakly compressible condition given below.

Definition 2.3 (Weakly compressible problems). Let $D = (L_{\text{yes}}^D, L_{\text{no}}^D)$ be a promise problem. We say that the promise problem D is weakly compressible if for every $\alpha \in \mathbb{N}$, there exists an algorithm Compress_α^D , which takes, as input, $\langle \text{Seq}_D \rangle$, a description of a uniform instance of promise problems for D . Compress_α^D outputs a description for a uniform instance of promise problems for D , $\langle \text{Seq}_D^{\text{comp}} \rangle$, such that the following holds: There exists an integer $\gamma = O(\text{poly}(\alpha))$ such that

- (1) (Runtime): Compress_α^D runs in $O(\text{poly}(|\langle \text{Seq}_D \rangle|, \alpha))$ time.
- (2) (Consistency of the output): $\text{Seq}_D^{\text{comp}}(n) \in D$ for all $n \in \mathbb{N}$, even if the initial input for Compress_α^D is not a valid uniform instance of D .
- (3) (Complexity bound for the output): $\text{Seq}_D^{\text{comp}}$ runs in time $O(\log(n)^\gamma)$.

Furthermore, if there exists some constant $n_0 \in \mathbb{N}$ such that for all $n \geq n_0$, $\text{Seq}_D(n)$ runs in time less than n^α . Then there exist some constant $n_0^{\text{comp}} = \text{poly}(\gamma, n_0)$ such that for all $n \geq n_0^{\text{comp}}$

- (Completeness): $\text{Seq}_D(n) \in L_{\text{yes}}^D \implies \text{Seq}_D^{\text{comp}}(n) \in L_{\text{yes}}^D$.
- (Soundness): $\text{Seq}_D(n) \in L_{\text{no}}^D \implies \text{Seq}_D^{\text{comp}}(n) \in L_{\text{no}}^D$.

Intuitively, instead of requiring a single Compress^D , a decision problem is weakly compressible if for every $\alpha \in \mathbb{N}$, there exists a Compress_α^D which only “compresses” uniform instances with runtime $O(n^\alpha)$. Clearly, if D is compressible, the compression algorithm guaranteed by the compressible condition can be used to satisfy the condition for weakly compressible. Namely, we have the following theorem.

THEOREM 2.4. *If $D \leq \text{RE}$, then the following are equivalent:*

- (1) D is RE-complete.
- (2) D is compressible decision problem.
- (3) D is weakly compressible decision problem.

Furthermore, if $D \leq \text{coRE}$, the above statement is true except with clause 1 replaced by “ D is coRE-complete”.

This shows that this weaker notion of compressibility can also be used for showing RE/coRE-completeness. We describe the compressible condition and weakly compressible for promise problems in more detail in the full manuscript in [23, Chapter 4].

3 Applying the Compressible Condition to Non-Local Games

Recall from the previous section that the tensor product value problem is complete with respect to the complexity class MIP^* , and the commuting operator value problem is complete with respect to the complexity class MIP^{co} . A uniform problem instance for the tensor product/commuting operator value problem is defined as a Turing machine $\mathcal{V} : \mathbb{N} \rightarrow \mathcal{G}$, where to abuse notation, $\mathcal{G}$ in this case is the set of all possible descriptions for a non-local game. One could intuitively think of the uniform sequence as the “inputted Turing machine”

We show that a specific subclass of game sequences, which we refer to as “conditionally linear verifier” admits a gap compression theorem described earlier in the introduction. We give an informal description of the conditionally linear verifier in the next section and the more detailed version in the full manuscript in [23, Chapter 5]. We give an informal version of the gap compression theorem below.

THEOREM 3.1 (GAP COMPRESSION THEOREM). *For every $\alpha \in \mathbb{N}$, there exists a polynomial time algorithm $\text{Gapcompress}_\alpha$, that takes the input $\mathcal{V} : \mathbb{N} \rightarrow \mathcal{G}$ a conditionally linear verifier such that $\mathcal{V}(n)$ runs in $O(n^\alpha)$ time, each game in the sequence can be sampled and decided in $O(n^\alpha)$ time. The algorithm runs in $\text{poly}(|\mathcal{V}|, \alpha)$ time and outputs a conditionally linear verifier $\mathcal{V}^{\text{Comp}} : \mathbb{N} \rightarrow \mathcal{G}$ such that, for $t \in \{*, \text{co}\}$:*

- (1) (Runtime) $\mathcal{V}^{\text{Comp}}(n)$ runs in $O(\text{polylog}(n))$ time, and each game $\mathcal{V}^{\text{Comp}}(n)$ can be sampled in $O(\text{polylog}(n))$ time, and the decider function D_n runs in $O(\text{polylog}(n))$ time.
- (2) (Completeness) If $\omega^t(\mathcal{V}(n)) = 1$, then $\omega^t(\mathcal{V}^{\text{Comp}}(n)) = 1$
- (3) (Soundness) If $\omega^t(\mathcal{V}(n)) \leq \frac{1}{2}$, then $\omega^t(\mathcal{V}^{\text{Comp}}(n)) \leq \frac{1}{2}$.

Roughly speaking, by considering the $\text{Gapcompress}_\alpha$ guaranteed by the theorem, this shows that $\text{MIP}^*/\text{MIP}^{\text{co}}$, when restricted from games generated by a conditionally linear verifier, are weakly compressible. In practice, there are many more caveats to the above theorem which is needed to argue for weakly compressible which was not listed above. Instead, we refer to the full manuscript in [23, Chapter 6] for more details. In conjunction with the NPA-hierarchy [26], which is a coRE algorithm for the commuting operator value problem (i.e. it halts if the game is in the set L_{no} and otherwise runs forever), we can conclude the coRE-completeness of MIP^{co} , thus showing the main theorem in this paper.

On the other hand, by combining the Gap compression with the well-known Searchfrombelow algorithm, an RE algorithm for the

tensor product value problem, we give an alternative proof for the $\text{MIP}^* = \text{RE}$ theorem using the weakly compressible condition we described above.

We remark that the $\text{Gapcompress}_\alpha$ algorithm is a more streamlined version of [19, Theorem 12.1], and the conditionally linear verifier is a more concise version of the normal form verifier defined in [19, Definition 5.31]. The primary challenge in this paper is to show that the same compression algorithm also has the desired completeness/soundness condition under the commuting operator model, which we summarize in Section 5.

In this paper, we also model finding an explicit separation between the models of entanglement as the $\frac{1}{2}$ -Bell test separation decision problem, which we model as a decision problem over the following two sets of non-local games:

- $\text{L}_{\text{yes}} = \{\mathcal{G} : \omega^*(\mathcal{G}) = \omega^{\text{co}}(\mathcal{G})\}$.
- $\text{L}_{\text{no}} = \{\mathcal{G} : |\omega^*(\mathcal{G}) - \omega^{\text{co}}(\mathcal{G})| \geq \frac{1}{2}\}$.

We remark that the constant above being $\frac{1}{2}$ can be changed to any arbitrary fixed constant $c \in (0, 1)$ by increasing the number of parallel repetition. The problem is known to be in RE by combining Searchfrombelow together with the NPA hierarchy, since the $\text{Gapcompress}_\alpha$ algorithm given in Theorem 3.1 preserves the tensor product value and the commuting operator value. It is not hard to see that $\text{Gapcompress}_\alpha$ also preserves the yes/no case for any given conditionally linear verifier and thus can be used to argue that the above problem is weakly compressible. Although it is impossible to find a game that realizes the separation computationally, such a separation can still potentially be found using mathematical technique.

4 Towards Proving the Gap Compression Theorem

In this subsection, we provide, from an algorithmic level, a rough outline of the subroutine used in Gapcompress given in Theorem 3.1. We first give a high-level description of a conditionally linear verifier and some intuition as to why the runtime can be compressed. In standard non-local game literature, a non-local game is defined in terms of two finite sets, X for the question set, $\mathcal{A}$ for the answer set, $\mu \sim X \times X$ as the question distribution for the two provers and $D : X^2 \times \mathcal{A}^2 \rightarrow \{0, 1\}$ as the validation function. A conditionally linear verifier $\mathcal{V}$ is a game sequence that takes as input $n \in \mathbb{N}$ and outputs a non-local game $\mathcal{G}_n$ with the following properties:

- Each game in the sequence has a question distribution that follows a specific class of distributions known as “conditionally linear distributions”. Intuitively, a verifier can sample a question pair by first sampling a seed s and then applying two special deterministic functions L^A, L^B in order to compute the question pair $(\text{L}^A(s), \text{L}^B(s))$. [19] realizes that a pair of honest provers can sample a question pair (in a way such that the question sampled for one prover is unknown to the other prover) from the conditionally linear distribution by taking the outcome from a specific set of Pauli Z measurements on shared EPR pairs (where a single EPR pair corresponds to the state $\frac{|00\rangle + |11\rangle}{\sqrt{2}}$). Hence, the verifier can effectively shrink the question size for the non-local game by simply asking

honest provers to perform the correct measurement on some pre-prepared EPR pairs between the provers. As seen later in this section, this fact is used with self-testing techniques to force the provers to make the correct Z measurements.

- $\mathcal{V}$ is defined by a pair of Turing machines (Q, D) . The sampler, Q , takes as input a natural number n and returns a description for the two functions L_n^A, L_n^B which describes the conditionally linear distribution that samples a question pair for $\mathcal{G}_n$. The decider, D , also takes a natural number and returns a description of a Turing machine, which computes D_n , the decision function for $\mathcal{G}_n$. This definition is closer to a definition given in the interactive proof system literature, and this allows us to use techniques from the PCP theorem to shrink the answer size for the game.

The runtime, or the complexity for the conditionally linear verifier is defined as the maximum runtime for the Turing machine Q and D . We remark that although the question set and the answer set are not explicitly specified by $\mathcal{V}$ in a conditionally linear verifier, both Q and D being time bounded effectively define a finite set of questions/answers for each of the games $\mathcal{G}_n$ ³. The Gapcompress algorithm constructed for a conditionally linear verifier consists of three main subroutines: **question reduction**, **answer reduction**, and **parallel repetition**, in which we briefly summarized below. Since many of the techniques used are similar to the ones used in [19], we only give a brief summary of each subroutine and highlight our improvements over [19] below. For a more detailed summary, we instead refer the readers to [19, Chapter 2].

Question Reduction. The goal of the question reduction protocol is to force the provers to make an “ideal” measurement in order for them to sample from a question pair following the conditionally linear verifier. The question reduction is a combination of two tests: the Pauli Basis test and the Introspection protocol. The Pauli Basis test uses self-testing techniques in order to force the dishonest provers to prepare enough EPR pairs required to sample the input distribution, as well as perform an X or Z measurement on all the prepared EPR pairs. The Introspection protocol utilizes the EPR pairs guaranteed by the Pauli Basis test and forces the provers to make the correct Pauli Z measurement so that the provers can sample a pair of questions from the given conditionally linear distribution. The Pauli basis test has a question sampling complexity of $\text{polylog}(n)$, and the Introspection protocol has a sampling complexity independent of n (where both tests have a decision complexity of $\text{poly}(n)$), thereby reducing the complexity of Q from $\text{poly}(n)$ to $\text{polylog}(n)$.

In this paper, we use the recent simplification due to [12] for the Pauli Basis test, which circumvents the low-individual degree test subroutine used in the original Pauli Basis test [19, Section 7]. As a result, the increase in soundness error in the question reduction theorem proven in our paper is independent of the size of the game, an improvement over the polynomial dependency in [19].

Answer Reduction. The goal of the answer reduction protocol is to reduce the complexity of the decider D from $\text{poly}(n)$ time to $\text{polylog}(n)$ time while retaining a $\text{polylog}(n)$ runtime for the

sampler. Similarly to [19, 25], a classical probabilistically checkable proof (PCP) is used on the verification Turing machine D_n . In particular, we use the same tailor-made PCP procedure used within [19]’s answer reduction protocol as a black box in this paper. The PCP procedure reduces checking computation steps of D_n returning accept given the corresponding question/answer pair into checking whether the two provers share the same collection of low individual degree polynomials with specific properties, which can then be checked using the “quantum low-individual degree test” introduced in [20].

There is an immediate problem with this approach. Recall that for a classical PCP which verifies an NP instance, the provers are intuitively trying to prove the following statement to the verifier: “given $x \in L$ and a polynomial size circuit C , there exists a proof string s such that $C(x, s) = 1$. Where as in the non-local game setting, the provers are trying to prove: “given a validation function D_n for a non-local game, and a question pair (x, y) , there exists an answer pair (a, b) , which is generated by two entangled provers, such that $D_n(x, y, a, b) = 1$. In order to compute the second statement in a PCP instance, both provers need to somehow play the game using a predetermined entangled strategy and output their answer without communicating with each other. Then, the provers need to pass their answer so that they can encode the computation step of $D_n(x, y, a, b)$ as a PCP instance. In order to get around this issue, a transformation known as Oracularization is applied before computing the PCP instance, whereby the question set and the answer set can be safely passed to one of the provers by using the other prover as a “consistency checker”. To ensure completeness is preserved through the oracularization transformation, we need an additional property in the completeness statement in the gap compression theorem: The perfect strategies in the original game must use a special kind of strategy known as an *oracularizable strategy*. Roughly speaking, a strategy is oracularizable if the player’s measurement observable commute in the same Hilbert space whenever they are given a question pair (x, y) such that $\mu(x, y) > 0$. We remark that this transformation is also used in [19], and the oracularizable strategy in this paper is the same as the “commuting and consistent strategy” used in [19].

Parallel repetition. By applying the above two subroutines to a conditionally linear verifier $\mathcal{V} : \mathbb{N} \rightarrow \mathcal{G}$ with sampling and verification complexity $O(\text{poly}(n))$, the resulting conditionally linear verifier $\mathcal{V}' : \mathbb{N} \rightarrow \mathcal{G}$ which runs in $O(\text{polylog}(n))$ time, such that for $t \in \{*, co\}$

- (1) (Completeness): If $\omega^t(\mathcal{V}(n)) = 1$, then $\omega^t(\mathcal{V}'(n)) = 1$,
- (2) (Soundness): If $\omega^t(\mathcal{V}(n)) \leq \frac{1}{2}$, then

$$\omega^t(\mathcal{V}'(n)) \leq 1 - \text{polylog}(n).$$

Thus, in order to show Theorem 3.1, one would need to apply a logarithmic-fold parallel repetition transformation to the game in order to amplify the “soundness” condition for $\mathcal{V}'$ to $\leq \frac{1}{2}$ while retaining the “completeness” condition. Where recall, r -fold parallel repetition is a transformation for the game $\mathcal{G}$ in which r question pairs are sampled independently and sent to the provers, and the provers must treat each of the r question pairs as independent questions and reply with r corresponding answer pairs. The provers only win the r -fold parallel repetition game if they win on all

³if Q runs in time T , then Q can sample a string with length at most T , which means that the question set can effectively be taken as $\{0, 1\}^T$

r independent instances of the game. We remark that applying a logarithmic-fold parallel repetitions to $\mathcal{V}'$ only increases the runtime by a logarithm factor.

Unfortunately, a strong parallel repetition theorem, i.e. a parallel repetition theorem that shows an exponential decay in the optimal success rate for entangled provers is an open problem. However, [6] shows that by applying a simple transformation, the anchored transformation, the resulting game would have a strong parallel repetition theorem, and this version of parallel repetition has been used in [19]. We use a slight modification for the anchoring transformation⁴ in our paper.

5 From MIP^* to MIP^{co}

Finally, we discuss some of the challenges in extending the gap compression theorem to the commuting operator models. Since the commuting operator model of entanglement cannot be approximated by finite-dimensional strategies, unlike the tensor product model, many techniques used in [19] for proving the gap compression theorem are not known to generalize to the commuting operator model.

[22] recently introduced a subclass of (two prover) commuting operator strategies known as *tracially embeddable strategies*. Tracially embeddable strategies, while being infinite dimensional, have many similar structures to a finite-dimensional tensor product strategy. Hence, many techniques from [19], which hold for the finite-dimensional tensor product model, can easily be translated to the setting where the provers are restricted to tracially embeddable strategies.

Furthermore, [22] shows that the behaviour of provers with access to the commuting operator model of entanglement can be **well approximated** by provers restricted to using tracially embeddable strategies. To be a bit more precise, the set of *correlations*, or the set of probability distributions for outputting a certain answer pair given a question pair when the provers are given access to tracially embeddable strategies is dense within the set of correlations for provers with commuting operator strategies. Hence, the complexity of MIP^{co} is precisely the same as the complexity of an interactive proof system where the provers are restricted to using tracially embeddable strategies. By working within this class of strategy, the following techniques used in the proof of the gap compression theorem in [19] become available in the analysis of MIP^{co} :

- (1) Tracially embeddable strategies provide a natural generalization to “density matrices” and the “observable switching trick” to finite-dimensional strategies. This is used in [22] to replicate the rigidity statement for the Pauli basis test for provers restricted to using tracially embedded strategies similarly to [19, Theorem 7.14] (which shows the rigidity for finite dimensional tensor product strategies). This is a key part of showing the “soundness” properties of the question reduction protocol in the commuting operator model.
- (2) The answer reduction protocol relies on the “soundness” of the quantum low-individual degree test, which is only shown to hold for a special class of strategies known as synchronous strategies in [20]. In [31], a “rounding” lemma, or a lemma translating results proven using synchronous

strategies to regular strategies when restricted to finite dimensional strategies, was shown. By working with tracially embeddable strategies, the same lemma can be shown for tracially embeddable strategies in [22]. We remark that the “rounding” lemma can be proven without using the tracially embeddable strategies framework by the works of [13].

For more intuition about tracially embeddable strategies in the finite-dimensional setting, we refer to [22, Example 3.3].

6 Parallel Repetition in the Commuting Operator Model

We also give the first “informational-theoretic” proof of parallel repetition theorem for the commuting operator model in this work. We remark that prior to this work, the only known example of a class of game which exhibits an exponential parallel repetition theorem was the XOR game [10], in which the tensor product value and the commuting operator value coincide [30].

We show that the informational-theoretical tools used to proof the parallel repetition theorem for anchored games [6], such as an analogue of mutual information and Uhlmann’s theorem, have an appropriate analogue in the commuting operator model and hence the majority of the proof from [6] can be shown for the commuting operator model. This is proven by combine the relative entropy formulation given in [1] for states on von Neumann algebras with the Tracially embeddable strategy mentioned above. We believe these techniques also could be useful to show a parallel repetition theorem for other classes of games, and could potentially be of interest to the quantum information community.

The proof of the parallel repetition for anchored games in this paper emerged from an early collaboration with William Slofstra and Henry Yuen. For more details on the proof of the anchored parallel repetition theorem in the commuting model, we refer the readers to the full manuscript in [23, Appendix A]

Acknowledgments

I thank William Slofstra, Henry Yuen, and Hamoon Mousavi for their involvement in the parallel repetition project, which led to this work. I thank Henry Yuen and Thomas Vidick for many email exchanges for explaining the $\text{MIP}^* = \text{RE}$ result. I thank Seyed Sajjad Nezhadi for explaining the conjectured approach for showing $\text{MIP}^{\text{co}} = \text{coRE}$ in [24]. I thank Andrew Marks for showing me Theorem 2.2, and the discussion related to the compressible condition, I thank Michael Chapman for the discussion on Oracularization and Oracularizable strategies, and I thank Aareyan Manzoor for some of the discussion on the potential applications on operator algebras in this work. I thank Srijita Kundu for helpful discussion about the parallel repetition theorem. I thank Yuming Zhao for helpful discussion related to this work. I thank Nick Resch, William Slofstra, Ronald de Wolf, as well as anonymous reviewer for both QIP and STOC for going through the earlier draft of this paper. I thank the anonymous FOCS reviewer for the helpful comments. Finally, I would like to thank my supervisors, Stacey Jeffery and Jonas Helsen, for going through some of the earlier drafts of this paper and listening to me yapping about MIP^{co} for the past year. This work is supported by ERC STG grant 101040624-ASC-Q.

⁴The modification is designed to preserve synchronicity within the game.

References

- [1] Huzihiro Araki. 1977. Relative Entropy for States of von Neumann Algebras. II. *Publications of the Research Institute for Mathematical Sciences* 13, 1 (1977), 173–192. doi:10.2977/prims/1195190105
- [2] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. 1998. Proof Verification and the Hardness of Approximation Problems. *J. ACM* 45, 3 (May 1998), 501–555. doi:10.1145/278298.278306
- [3] Sanjeev Arora and Shmuel Safra. 1998. Probabilistic Checking of Proofs: A New Characterization of NP. *J. ACM* 45, 1 (Jan. 1998), 70–122. doi:10.1145/273865.273901
- [4] Jananan Arulseelan and Aareyan Manzoor. 2025. The Universal Theory of Locally Universal Tracial von Neumann Algebras Is Not Computable. arXiv:2508.21709 [math] doi:10.48550/arXiv.2508.21709
- [5] László Babai, Lance Fortnow, and Carsten Lund. 1991. Non-Deterministic Exponential Time Has Two-Prover Interactive Protocols. *computational complexity* 1, 1 (March 1991), 3–40. doi:10.1007/BF01200056
- [6] Mohammad Bavarian, Thomas Vidick, and Henry Yuen. 2021. Anchored Parallel Repetition for Nonlocal Games. arXiv:1509.07466 [quant-ph] (March 2021). arXiv:1509.07466 [quant-ph]
- [7] J. S. Bell. 1964. On the Einstein Podolsky Rosen Paradox. *Physics Physique Fizika* 1, 3 (Nov. 1964), 195–200. doi:10.1103/PhysicsPhysiqueFizika.1.195
- [8] Joseph Bowles, Ivan Šupić, Daniel Cavalcanti, and Antonio Acín. 2018. Self-Testing of Pauli Observables for Device-Independent Entanglement Certification. *Physical Review A* 98, 4 (Oct. 2018), 042336. arXiv:1801.10446 [quant-ph] doi:10.1103/PhysRevA.98.042336
- [9] R. Cleve, P. Hoyer, B. Toner, and J. Watrous. 2004. Consequences and Limits of Nonlocal Strategies. In *Proceedings. 19th IEEE Annual Conference on Computational Complexity, 2004*. 236–249. doi:10.1109/CCC.2004.1313847
- [10] Richard Cleve, William Slofstra, Falk Unger, and Sarvagya Upadhyay. 2008. Strong Parallel Repetition Theorem for Quantum XOR Proof Systems. arXiv:quant-ph/0608146 doi:10.48550/arXiv.quant-ph/0608146
- [11] A. Connes. 1976. Classification of Injective Factors Cases. *The Annals of Mathematics* 104, 1 (July 1976), 73. jstor:1971057 doi:10.2307/1971057
- [12] Mikael de la Salle. 2022. Spectral Gap and Stability for Groups and Non-Local Games. arXiv:2204.07084 [math]
- [13] Mikael de la Salle and Amine Marrakchi. 2023. Almost Synchronous Correlations and Tomita-Takesaki Theory. (Oct. 2023). doi:10.48550/arXiv.2307.08129
- [14] Artur K. Ekert. 1991. Quantum Cryptography Based on Bell's Theorem. *Physical Review Letters* 67, 6 (Aug. 1991), 661–663. doi:10.1103/PhysRevLett.67.661
- [15] Tobias Fritz, Tim Netzer, and Andreas Thom. 2014. Can You Compute the Operator Norm? *Proc. Amer. Math. Soc.* 142, 12 (Aug. 2014), 4265–4276. arXiv:1207.0975 [math] doi:10.1090/S0002-9939-2014-12170-8
- [16] Isaac Goldbring and Thomas Sinclair. 2025. On Definability of C*-Tensor Norms. arXiv:2509.15086 [math] doi:10.48550/arXiv.2509.15086
- [17] Rudolf Haag and Daniel Kastler. 1964. An Algebraic Approach to Quantum Field Theory. *J. Math. Phys.* 5, 7 (July 1964), 848–861. doi:10.1063/1.1704187
- [18] Rahul Jain, Carl A. Miller, and Yaoyun Shi. 2020. Parallel Device-Independent Quantum Key Distribution. *IEEE Transactions on Information Theory* 66, 9 (Sept. 2020), 5567–5584. arXiv:1703.05426 [quant-ph] doi:10.1109/TIT.2020.2986740
- [19] Zhengfeng Ji, Anand Natarajan, Thomas Vidick, John Wright, and Henry Yuen. 2022. MIP* = RE. arXiv:2001.04383 [quant-ph] doi:10.48550/arXiv.2001.04383
- [20] Zhengfeng Ji, Anand Natarajan, Thomas Vidick, John Wright, and Henry Yuen. 2022. Quantum Soundness of Testing Tensor Codes. arXiv:2111.08131 [quant-ph]
- [21] Neil D. Jones. 1997. *Computability and Complexity: From a Programming Perspective*. MIT press, Cambridge (mass.).
- [22] Junqiao Lin. 2024. Tracial Embeddable Strategies: Lifting MIP* Tricks to MIPco. arXiv:2304.01940 [quant-ph] doi:10.48550/arXiv.2304.01940
- [23] Junqiao Lin. 2025. MIPco = coRE. arXiv:2510.07162 [quant-ph] doi:10.48550/arXiv.2510.07162
- [24] Hamoon Mousavi, Seyed Sajjad Nezhadi, and Henry Yuen. 2022. Nonlocal Games, Compression Theorems, and the Arithmetical Hierarchy. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing (STOC 2022)*. Association for Computing Machinery, New York, NY, USA, 1–11. doi:10.1145/3519935.3519949
- [25] Anand Natarajan and John Wright. 2019. NEXP in MIP*. arXiv:1904.05870 [quant-ph]
- [26] Miguel Navascués, Stefano Pironio, and Antonio Acín. 2008. A Convergent Hierarchy of Semidefinite Programs Characterizing the Set of Quantum Correlations. *New Journal of Physics* 10, 7 (July 2008), 073013. arXiv:0803.4290 [quant-ph] doi:10.1088/1367-2630/10/7/073013
- [27] Seyed Sajjad Nezhadi, Andrew Marks, and Henry Yuen. 2025. The Recursive Compression Method for Proving Undecidability Results. *In preparation* (2025).
- [28] Narutaka Ozawa. 2013. About the Connes Embedding Conjecture—Algebraic Approaches—. arXiv:1212.1700 [math] doi:10.48550/arXiv.1212.1700
- [29] Michael Sipser. 2006. *Introduction to the Theory of Computation*. Thomson Course Technology.
- [30] Boris Tsirelson. 1987. Quantum Analogues of the of the Bell Inequality. *Journal of Soviet Mathematics* 36 (Feb. 1987), 557–570.
- [31] Thomas Vidick. 2022. Almost Synchronous Quantum Correlations. *J. Math. Phys.* 63, 2 (Feb. 2022), 022201. arXiv:2103.02468 doi:10.1063/5.0056512

Received 2025-10-27; accepted 2026-02-01