# Quantum Cryptographic Developments Stemmed from Proof Systems

Mark Chen

15 April 2024

## An Introduction: QMA and Local-Hamiltonian

## 1 Quantum Merlin-Arthur

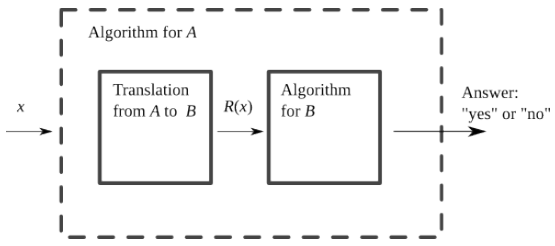
### 1.1 Classical Motivation

In classical complexity, non-deterministic polynomial time (**NP**) is a class of decision problems that can be verified in polynomial time. Decision problems are problems where each language $L$ is split into yes (1) and no (0) answers $L = (L_1, L_0)$.

**Definition 1.** *A decision problem $L = (L_1, L_0) \in$ **NP** iff there exists a deterministic verifier algorithm $V(x, y)$ such that*

1. *(Efficiently Verifiable) $A$ runs in poly-time with respect to $n = |x|$.*
2. *(Completeness) If $x \in L_1$, there exists a string $y \in \{0,1\}^{p(n)}$ such that $V(x, y) = 1$.*
3. *(Soundness) If $x \in L_0$, $\forall y \in \{0,1\}^{p(n)}$, $A(x, y) = 0$.*

**Definition 2** (**NP**-Completeness)**.** *Any* **NP** *problem can be poly-time reduced to any* **NP***-complete problem ($\exists$ a poly-time deterministic algorithm that maps all YES/NO instances of the original problem to the* **NP***-complete problem so that solving the complete problem solves the original problem, up to the difference of a poly-time run-time factor).*




**Example 1** (**NP**-Complete Problems)**.** *The most famous such problem is* 3-SAT *(proved by Cook-Levin Theorem). Other examples include* Traveling-Sales-Person *(*TSP*) and* 0/1-Integer-Programming*. At this point there are tens of thousands of* **NP***-complete problems that imply various applications.*

### 1.2 Quantum Analogy of NP

**QMA** is defined by slackening the conditions for **NP** to allow probabilistic in the process.

**Definition 3** (**QMA**). *A promise problem $L = (L_1, L_0, L_*)$ is in* $\mathbf{QMA}(b, a)$ *iff there exists a uniform family $\{C_n\}$ of poly-size quantum circuits that take in two input registers, $x \in \{0,1\}^n$, and output a single qubit:*

1. *(Completeness) If $x \in L_1 \cap \{0,1\}^n$, then $\exists p(n)$-qubit state $|\psi\rangle$ s.t.*

$$\Pr[C_n(x, |\psi\rangle) = 1] \geq b.$$

2. *(Soundness) If $x \in L_1 \cap \{0,1\}^n$, then $\forall p(n)$-qubit state $|\psi\rangle$ s.t.*

$$\Pr[C_n(x, |\psi\rangle) = 1] \leq a.$$

**Proposition 1.** *Generally, the class* **QMA** *is defined as* $\mathbf{QMA}(2/3, 1/3)$*, but we can prove that for $b - a \geq 1/poly(n)$,* $\mathbf{QMA}(b, a) = \mathbf{QMA}(2/3, 1/3)$.

*Proof.* The proof is the same as the classical **BPP** amplification proof, which states that as long as there is an inverse polynomial gap between the accepting and rejecting probabilities, one can always amplify the probability to be equivalent to $(2/3, 1/3)$.

One thing that was thought to also be different is that such a naive approach would actually increase the witness-size by a factor of $O(\log(1/\delta))$ as well, which is not ideal. [MW05] came up with a surprising yet beautiful approach that, instead, increases the run-time by a factor of $O(\log(1/\delta))$ and leaves the length of the witness being $p(n)$ qubits. ■

**Remark 1.** *What is special about* **QMA** *compared to* **NP** *and* **BPP***? The quantum power comes at the properties that:*

1. *(Quantum Witness): The witness is a quantum state, $|\psi\rangle$.*
2. *(Quantum Verifier): The verifier algorithm is a circuit that takes in two quantum registers and and outputs a qubit.*

*We call the special case where we take out the first property and make the witness classical bit string the* **QCMA** *class ("C" for classical). Then, we call the special case where we make the verifier algorithm classical* **MA** *(this step cannot be done before the first special case, because as long as the witness is still quantum it doesn't make sense to run it by a classical verifier).*

**Proposition 2.** *By the chain of special cases in remark 1, it is easy to see that*

$$\mathbf{NP} \subseteq \mathbf{MA} \subseteq \mathbf{QCMA} \subseteq \mathbf{QMA}$$

**Conjecture 1.** *It is conjectured that* $\mathbf{MA} = \mathbf{NP}$ *and* $\mathbf{NP} \subsetneq \mathbf{QMA}$*. In other words, it is believed that "quantum proofs" can efficiently prove more than "classical proofs" can.*

## 1.3 Open Problem

The following problem is in **QMA**, but it is unknown if it is **QMA**-complete and unknown if it is in **NP**.

**Definition 4.** *The group non-membership problem. Consider a finite group $G$ defined by its generator, subgroup $H \leq G$, and element $g \in G$. $(L_1)$ $g \notin H$. $(L_0)$ $g \in H$. First of all, notice that it should be easy to show that group membership problem is easily in* **NP***. Now, consider this problem, the verifier is the superposition of all states in $H$, and all you need to show that one of them is indeed $g$.*

# 2 QMA Complete Problems

## 2.1 Physics: The Hamiltonian

In a molecular/cluster system involving $n$ electrons, the energy of these electrons is the sum of their kinetic energy $\hat{T}$, their electric potential energy from atomic nuclei $\hat{V}$, and electron-electron repulsion energy $\hat{U}$. The Hamiltonian for this system is

$$\hat{H} = \hat{T} + \hat{V} + \hat{U}. \tag{1}$$

In general, the Hamiltonian is the sum of all the energy operators of a system.

All observables $\hat{O}$ (position, momentum, energy, etc.) of a state $|\psi\rangle$ are described by operators in quantum mechanics. The observed value is given by the expectation value of the operator with the state

$$\langle\psi|\,\hat{O}\,|\psi\rangle\,. \tag{2}$$

When $|\lambda\rangle$ is a normalized eigenvector of $\hat{O}$ with eigenvalue $\lambda$,

$$\langle\lambda|\,\hat{O}\,|\lambda\rangle = \lambda. \tag{3}$$

Since all observables are real-valued in the physical world, we constrain all eigenvalues of observables to be real. This can be done by enforcing $\hat{O}$ to be Hermitian ($\hat{O} = \hat{O}^\dagger$, where $\dagger$ represents the conjugate transpose). Another neat consequence of making observables Hermitian is that the eigenvectors of $\hat{O} : \mathbb{C}^d \to \mathbb{C}^d$ span the full rank $d$ of the space $\mathbb{C}^d$ (see spectral theorem of Hermitian matrices). This extends to $d \to \infty$. More specifically, we can diagonalize the Hamiltonian is

$$\hat{H} = \sum_n E_n\,|n\rangle\,\langle n|\,, \tag{4}$$

where $|n\rangle$ are the eigenvectors of $\hat{H}$ with eigenvalue $E_n$. The energy of an arbitrary state can thus be written as the following expectation value

$$\langle\psi|\,\hat{H}\,|\psi\rangle = \langle\psi|\left(\sum_n E_n\,|n\rangle\,\langle n|\right)|\psi\rangle \tag{5}$$

$$= \sum_n E_n\,\langle\psi|n\rangle\,\langle n|\psi\rangle \tag{6}$$

$$= \sum_n E_n||\,\langle\psi|n\rangle\,||^2. \tag{7}$$

To maximize entropy per the second law of thermodynamics, at thermal equilibrium, a system distributes its states according to the probability distribution

$$Pr(|\psi\rangle = |n\rangle) \propto \exp\left\{-\frac{E_n}{k_B T}\right\}, \tag{8}$$

meaning the lowest-energy state $E_0$, often called the ground state, has the highest probability of occurring. If we cool the system such that $T \to 0$, it is the only state that will exist.

In physics, we generally care about time-independent eigenstates, though knowing these states allows us to determine the time-evolution of any arbitrary state. By Schrödinger's equation,

$$\hat{H}\,|\psi(t,\vec{x})\rangle = i\hbar\frac{\partial}{\partial t}\,|\psi(t,\vec{x})\rangle\,. \tag{9}$$

the time-evolution of a system depends on the Hamiltonian. The Hamiltonian is generally has no explicit time dependence (e.g. the electric potentials between two charged objects depends only on their position), so it makes some sense to separate the state into time and position components.

$$\psi(t, \vec{x}) = \phi(t)\psi(\vec{x}). \tag{10}$$

Let $|n(\vec{x})\rangle$ be an eigenvector of $\hat{H}$ with corresponding eigenvalue $E$.

$$\hat{H}\psi(t, \vec{x}) = E\phi_n(t) \left|n(\vec{x})\right\rangle. \tag{11}$$

The letter $E$ stands for energy. Plugging this into Schrödinger's equation gives

$$i\hbar\frac{\partial}{\partial t}\phi_n(t) \left|n(\vec{x})\right\rangle = E\phi_n(t) \left|n(\vec{x})\right\rangle \tag{12}$$

$$\phi_n(t) = \exp\left\{-i\frac{E}{\hbar}t\right\}. \tag{13}$$

Thus, the time-evolution of the eigenstate is described as a simple oscillatory function. At any moment in time $t = T$, any arbitrary state $|\Psi(t = T, \vec{x})\rangle$ can be represented as a sum of the eigenstates of the Hamiltonian as the eigenvectors span the full rank. If at $t = 0$,

$$|\Psi(0, \vec{x})\rangle = \sum_n \alpha_n \left|n(\vec{x})\right\rangle \tag{14}$$

for arbitrary weights $\alpha_n$, the time-evolved state is just

$$|\Psi(t, \vec{x})\rangle = \sum_n \alpha_n\phi_n(t) \left|n(\vec{x})\right\rangle \tag{15}$$

$$= \sum_n \alpha_n \exp\left\{-i\frac{E_n}{\hbar}t\right\} \left|n(\vec{x})\right\rangle. \tag{16}$$

### 2.1.1 The Classical-Ising Model

Minimization of Hamiltonians can be used to solve classical problems.

Let $G = (V, E)$ be a graph with $n$ verticies in set $V$ and pairs of verticies in the set $E$ of edges. Define $Z_i$ to be the Pauli $z$-matrix acting on qubit $i$. Then, the Max-Cut Hamiltonian is

$$\hat{H} = \sum_{e=(u,v)\in E} Z_u \otimes Z_v \tag{17}$$

$$= \sum_{(u,v)\in E} |00\rangle \langle 00|_{u,v} - |01\rangle \langle 01|_{u,v} - |10\rangle \langle 10|_{u,v} + |11\rangle \langle 11|_{u,v} \tag{18}$$

$$= \sum_{x\in\{0,1\}^n} (m - 2c(x)) \left|x\right\rangle \left\langle x\right|, \tag{19}$$

where $c(x)$ is the number of satisfied constraints (number of edges that will be cut) and $m = |E|$.

This is similar to the real-world quantum Heisenburg model subject to a transverse magnetic field on the $x$-axis. For some number of qubits arranged in a ring, the Hamiltonian for this model is

$$H(J_x, J_y, J_z, g) = J_x \sum_i X_i \otimes X_{i+1} + J_y \sum_i Y_i \otimes Y_{i+1} + J_z \sum_i Z_i \otimes Z_{i+1} + g \sum_i X_i, \tag{20}$$

where $J_x, J_y, g = 0$ gives the Max-Cut problem Hamiltonian and $J_x = J_y = J_z, g = 0$ gives something called the quantum Max-Cut problem. Minimizing the 2D extension to the Heisenburg Hamiltonian is **QMA**-complete.

## 2.2 The Local Hamiltonian Problem

**Definition 5.** *A $k$-local $n$-qubit Hamiltonian $\hat{H}$ acts trivially on only $k \leq n$ qubits. In other words, the $(2^n \times 2^n)$-dimensional $\hat{H}$ can be fully described as the tensor product of a $(2^k \times 2^k)$-dimensional tensor and identity for the other $n-k$ qubits.*

**Definition 6.** *The $k$-local Hamiltonian problem for a $n$-qubit Hamiltonian is a promise problem where (i) the Hamiltonian satisfies*

$$\hat{H} = \sum_{i=1}^{m} \hat{H}_i, \tag{21}$$

*where each $\hat{H}_i$ are $k'$-local where $k' \leq k$ and $0 \preccurlyeq \hat{H}_i \preccurlyeq \hat{I}_{2^n}$, and (ii) given $a, b \in [0, m]$ with $b - a \geq poly(n)$, decide whether the minimum eigenvalue $\lambda_0$ of $\hat{H}$ is $\leq a$ or $\geq b$ promised it is in one of the two categories.*

- *The accepted languages $L_1$ are the set of Hamiltonians where $\lambda_0 \leq a$.*
- *The rejected are those with $\lambda_0 \geq b$, and we are promised we do not get $L_*$, where $a < \lambda_0 < b$.*

In general, $\hat{H}_i$ may satisfy the condition $0 \preccurlyeq \hat{H} \preccurlyeq \hat{I}_{2^n}$. However, we can always normalize $\hat{H}$ such that the $-I_{2^n} \preccurlyeq \hat{H}_i \preccurlyeq I_{2^n}$. Then, we can define a new positive-semidefinite Hamiltonian $\hat{H}'_i = (\hat{H} + I_{2^n})/2$ that satisfies $0 \preccurlyeq \hat{H}'_i \preccurlyeq I_{2^n}$. Under this transformation, the eigenvalue $\lambda'_0$ of $\hat{H}' = \sum_{i=1}^{m} \hat{H}'_i$ is given by $\lambda'_0 = (\lambda_0 + m)/2$.

**Theorem 1.** *There is a version of the $k$-local Hamiltonian problem that is* **QMA***-complete.*

*Proof.* It is easy to verify that the $k$-local Hamiltonian problem is in **QMA**. To show completeness, for a promise problem $L$ in **QMA**, we can construct a map from all instances $x \in L$ to a Feynman-Kitaev Hamiltonian. For a given verifier circuit $C_n$, the input is the $n$-qubit string $x$, the $s$-qubit ancilla, and the $w(n) \leq \text{poly}(n)$ verifier. An additional $T + 1 \leq \text{poly}(n)$ time-keeping states are added for each gate $\{U_t\}_{t=1}^{T}$ where $C_n = U_T \cdots U_1$.

At the initial state, a penalty Hamiltonian

$$\hat{H}_{init} = \sum_{i=1}^{s} |1\rangle\langle 1|_{A,i} \otimes |0\rangle\langle 0|_C \tag{22}$$

checks to make sure the ancilla (denoted $A$) is initialized to zero and the proper input $x$ (denoted $X$) is given. The clock (denoted $C$) state is $|t = 0\rangle$.

Then, an additional $T$ Hamiltonians ($t \in \{1, \cdots, T\}$) assign penalties for deviating from the circuit

$$\hat{H}_t = \frac{1}{2}\left(I \otimes (|t-1\rangle\langle t-1|_C + |t\rangle\langle t|_C) - U_t \otimes |t\rangle\langle t-1|_C - U_t^* \otimes |t-1\rangle\langle t|_C\right). \tag{23}$$

Finally, a penalty is assigned for a 0-measurement in the output (treat a 1-measurement as accepting)

$$\hat{H}_{fin} = |0\rangle\langle 0|_1 \otimes |T\rangle\langle T|_C. \tag{24}$$

The total Feynman-Kitaev (penalty) Hamiltonian is just

$$\hat{H} = \hat{H}_{init} + \sum_{t=1}^{T} \hat{H}_t + \hat{H}_{fin}. \tag{25}$$

Completeness and soundness can be shown for this Hamiltonian with well-defined $a$ and $b$. Furthermore, as each quantum gate acts on at most 2 qubits and the clock register uses $\lceil \log(T+1) \rceil$ bits, we have $k$-locality where $k = \lceil \log(T+1) \rceil + 2 = O(\log(n))$. This can be further reduced to a constant. [dW23] ■

## 2.3 Additional QMA-Complete Problems

**Definition 7.** *The non-identity check problem. Given an n-qubit polynomial circuit $C$, determine whether this circuit is non-trivial up to a phase. ($L_1$) For all $\phi \in [0, 2\pi)$, $||C - e^{i\phi} I_{2^n}|| \geq b$. ($L_0$) There is some $\phi \in [0, 2\pi)$ such that $||C - e^{i\phi} I_{2^n}|| \leq a$. (Promise) Either $L_1$ or $L_0$ is the case and $b - a \geq 1/poly(n)$.*

**Definition 8.** *The k-local matrix consistency problem. Consider $m \leq poly(n)$ density matrices $\{\rho_i\}_{i=1}^m$, where $\rho_i$ depends only on the set of qubits $Q_i$ with $|Q_i| \leq k$. Denote $Q = \{1, \cdots, n\}$ as the set of all qubits. ($L_1$) There is a consistent n-qubit density matrix $\rho$, meaning for all $i \in \{1, \cdots, m\}$, the partial trace $tr_{Q \setminus Q_i}(\rho) = \rho_i$. ($L_0$) All matrices $\rho$ have a significant non-consistency, meaning there exists some $i \in \{1, \cdots, m\}$ such that $|tr(tr_{Q \setminus Q_i}(\rho) - \rho_i)| \geq b$. (Promise) Either $L_1$ or $L_0$ is the case and $b \geq 1/poly(n)$.*

# Beyond MA & QMA

This part of the talk is somewhat more advanced that involves the ideas of **QMA** that we have just introduced. This talk was inspired by Tina Zhang, who gave a great talk on this topic at QuACC a few Fridays ago on "compiled nonlocal games with applications in cryptography." We introduce some of the critical concepts here. We go through the following chain of generalizations:

$$\underset{\mathbf{QMA}}{\mathbf{MA}} \to \underset{\mathbf{QMA}(i)}{\mathbf{MA}(i)} \to \underset{\mathbf{QIP}}{\mathbf{IP}} \to \mathbf{MIP} \to \mathbf{MIP}^*.$$

## 1 $\underset{\mathbf{QMA}}{\mathbf{MA}} \to \underset{\mathbf{QMA}(i)}{\mathbf{MA}(i)}$

The reason why **QMA** is a quantum analogue of **NP** is really due to the fact that it is a quantum analogue of **MA**. Compare the following two definitions:

- ($L \in \mathbf{NP}$) "$x \in L \iff \exists y$, such that $V(x,y) = 1$ where $V$ runs in poly-time."
- ($L \in \mathbf{MA}$) "$x \in L \implies \exists y$, such that $\Pr_z [V(x,y,z) = 1] \geq \frac{2}{3}$ where $V$ runs in poly-time (clearly we also need the $x \notin L$ case)."

**Remark 2.** *Notice that* **MA** *is only different from* **NP** *in that it allows probabilistic power. This should explain why the conjecture 1 formulates that* $\mathbf{MA} = \mathbf{NP}$ *just like how it is conjectured that* $\mathbf{P} = \mathbf{BPP}$.

In fact, this probabilistic power could be derandomized with additional power, due to the following theorem:

**Theorem 2.** *We know that* **MA** *has a probabilistic* Verifier*, so are* $\mathbf{AM}[k]$ *for different constant values of* $k$*. However, it is actually the case that, given any of these, we can construct an equivalent protocol with* Verifier *making deterministic decisions.*

*Proof.* We start with **MA** as it is essentially the base case. Suppose the specific **MA** protocol that decides for a language $L$. Let $n$ be the input size, and $l = |r|$ which is the random coin flip size (which should be $\mathsf{poly}(n)$). Thus, by definition,

- If $x \notin L$, then $\forall m \in \{0,1\}^{p(n)}, \Pr_r [V(x,m,r) = 1] \leq \frac{1}{4l}$.
- If $x \in L$, then $\exists m \in \{0,1\}^{p(n)}, \Pr_r [V(x,m,r) = 1] \geq 1 - \frac{1}{4l}$.

Given $x$ and some $m$, denote the set of coin flips, $r$, that make $V(x,m,r)$ evaluate to 1 as $S_x^m$. We can show that, with a specific kind of amplification (through permutation by $\oplus$ [XOR]), there are the following case:

- (If $x \notin L$) $\forall$ set of permutations, none would cover the entire $\{0,1\}^l$, meaning it can find something to reject, perfectly. Specifically, what we want to show is:

$$|S_x^m| \leq \frac{1}{4l} 2^l \implies \forall z_1, \ldots, z_l, \bigcup_{i=1}^{l} S_x^m \oplus z_i \subsetneq \{0,1\}^l.$$

*Proof.* Observe that

$$\left| \bigcup_{i=1}^{l} S_x^m \oplus z_i \right| \overset{\text{union bnd}}{\leq} \sum_{i=1}^{l} (S_x^m \oplus z_i) \leq \sum_{i=1}^{l} (S_x^m),$$

because XOR can be seen as shifting the original set, so it could at most keep the size of the original set the same (if part of the original set got shifted outside of the entire domain, it may decrease the original set size).

$$\left| \bigcup_{i=1}^{l} S_x^m \oplus z_i \right| \leq \sum_{i=1}^{l} (S_x^m) \leq l \cdot |S_x^m| \leq l \frac{1}{4l} 2^l = \frac{2^l}{4} < 2^l.$$

■

- (If $x \in L$) We show that there exists some shift that $V$ can accept with perfect completeness, i.e.

$$|S_x^m| \geq \left(1 - \frac{1}{4l}\right) 2^l \implies \exists z_1, \ldots, z_l, \bigcup_{i=1}^{l} S_x^m \oplus z_i = \{0,1\}^l.$$

*Proof.* Observe that, fixing any $y \in \{0,1\}^l$,

$$\Pr_{z_1,\ldots,z_l}\left[y \notin \bigcup_{i=1}^{l} S_x^m \oplus z_i\right] \leq \Pr_{z_1}\left[y \notin S_x^m \oplus z_1\right] \cdots \Pr_{z_l}\left[y \notin S_x^m \oplus z_l\right] = \prod_{i=1}^{l} \Pr_{z_i}\left[y \notin S_x^m \oplus z_i\right],$$

but shifting by $z_i$ to not include $y$ is the same as shifting by $y$ to not include $z_i$, so

$$\Pr_{z_1,\ldots,z_l}\left[y \notin \bigcup_{i=1}^{l} S_x^m \oplus z_i\right] \leq \prod_{i=1}^{l} \Pr_{z_i}\left[y \notin S_x^m \oplus z_i\right] = \prod_{i=1}^{l} \Pr_{z_i}\left[z_i \notin S_x^m \oplus y\right] \leq \left(\frac{1}{4l}\right)^l.$$

Now, the probability such that such a $y$ exists to not be contained in such a shift is:

$$\Pr_{z_1,\ldots,z_m}\left[\exists y, y \notin \bigcup_{i=1}^{l} S_x \oplus z_i\right] \leq 2^l \cdot \left(\frac{1}{4l}\right)^l,$$

which means that, to get the probability that all $y$ would be covered is to take the complement of this, and that would give a probability of

$$1 - \frac{2^l}{(4l)^l} > 0 \implies \text{ such } z_1, \ldots, z_m \text{ exist by probabilistic method.}$$

■

But this is it, as we can reformulate the accepting and rejecting conditions in the $\Sigma_2^P \cap \Pi_2^P$ forms:

- $x \notin L \iff \forall m \in \{0,1\}^{p(n)}, z_1, \ldots, z_l, \exists y \in \{0,1\}^l$ s.t. $V(m, z_1, \ldots, z_l, y) = 1$ (which happens exactly when $y \notin \bigcup_{i=1}^{l} S_x^m \oplus z_i$).
- $x \in L \iff \exists m \in \{0,1\}^{p(n)}, z_1, \ldots, z_l, \forall y \in \{0,1\}^l$ s.t. $V(m, z_1, \ldots, z_l, y) = 1$ (which happens exactly when $y \in \bigcup_{i=1}^{l} S_x^m \oplus z_i = \{0,1\}^l$).

New protocol with completeness (so it must accept all $x \in L$):

1. Prover finds the $m$ and $z_1, \ldots, z_l$ and sent to Verifier.
2. Verifier uniformly randomly choose $y$ and check if $V(m, z_1, \ldots, z_l, y) = 1$. If so, accept; reject, otherwise (notice that, though it can still accept $x \notin L$ case which we will need to deal with using the soundedness protocol instead [which also rejects perfectly], the Verifier is capable of accepting every instance of $x \in L$ perfectly).

■

**Remark 3.** *Again,* **QMA** *and* **MA** *are only different, literally, in that* **QMA** *takes a witness as a $p(n)$-qubit state and its verifier can be a quantum circuit.*

## 2 $\mathbf{MA}(i) \to \mathbf{IP}$ / $\mathbf{QMA}(i) \to \mathbf{QIP}$

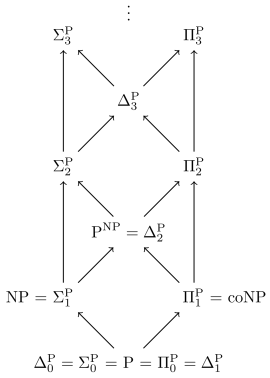
To get a sense of the full power of **IP**, let's first introduce the polynomial hierarchy.

## 2.1 Polynomial Hierarchy

Notice that **NP** language can be characterized by a single existential statement (described as the existence of a poly-size witness). It turns out this can be a lot more generalized to be the following two kinds of classes:

- $(L \in \Pi_i^p)\ x \in L \iff \forall y_1, \exists y_2, \ldots \exists / \forall y_i, V(x, y_1, y_2, \ldots, y_i) = 1.$
- $(L \in \Sigma_i^p)\ x \in L \iff \exists y_1, \forall y_2, \ldots \forall / \exists y_i, V(x, y_1, y_2, \ldots, y_i) = 1.$

This generalization will fill up the entire polynomial hierarchy, denoted **PH**, which looks like:




**Definition 9** (Totally Quantifiable Boolean Function (TQBF))**.** *This problem should capture the form of each specific instance of* **PH***, as it has the following form:*

$$Q_1 x_1, Q_2 x_2, \ldots, Q_n x_n, \phi(x_1, \ldots, x_n),$$

*where each $Q_i$ and $Q_{i+1}$ are alternating between $\exists$ and $\forall$, starting from either $\exists$ or $\forall$. $\phi$ is a Boolean formula with $x_1, \ldots, x_n$ as variables. The goal is to decide if $\phi(x_1, \ldots, x_n)$ is satisfiable.*

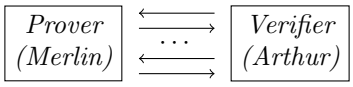
**Proposition 3.** *Any $\Pi_i^p$ and $\Sigma_i^p$ can be reduced to* TQBF*. This implies that* **PH** $\subseteq$ **PSPACE** *(it is not presently known if the equality holds).*

**Proposition 4.** TQBF *is* **PSPACE***-complete.*

## 2.2 Interactive Proof System (IP)

**Definition 10** (**MA**(i))**.** *Notice that what* **MA** *really did is a protocol that captures the probabilistic version of a $\Pi_1^p$ problem. So, we can easily generalize that to a bigger $i$, by letting there be multiple rounds of communications between Merlin (the all-powerful prover) and Arthur (the computationally limited verifier). That generalization, is what we call* **MA**$(i)$*.*

**Definition 11** (**IP**)**.** *Is the union of all* **MA**(poly$(n)$)*, plus it can be both directions (i.e. both the prover and the verifier can be the one to start the first talk).*




**Remark 4** ([GS86])**.** *This is not exactly true. Because the first motivation for* **IP** *to be introduced is the following:*

- *Consider the protocol for* Graph-Non-Isomorphism*. Suppose the input is $G_0, G_1$ such that they are not isomorphic to each other and the* Prover *is to convince the* Verifier *that they are not isomorphic:*

- VERIFIER *picks an* $b \in \{0, 1\}$.
- VERIFIER *randomly permutes* $G_b$ *and send to* PROVER.
- PROVER *is supposed to find the* $b$ *and send* $b$ *back.*

*The completeness and soundedness are both easy to check: because* VERIFIER *is all powerful, so, as long as* $G_0 \ncong G_1$*, there is only one* $G_b$ *that the permuted graph could be isomorphic to and so* VERIFIER *can always find it; but when* $G_0 \cong G_1$*, the best* VERIFIER *could do is to guess uniform randomly what* $b$ *is.*

- *Notice that the reason why this could be done is because* $b$*, the random coin flip of the* VERIFIER *is hidden from the* PROVER*. In fact, this is the key difference between* $\mathbf{AM}[k]$ *and* $\mathbf{IP}[k]$*, which is that* **AM** *is said to be the public-coin interactive proof system, and* **IP** *is said to be the private-coin interactive proof system.*
- *As it turned out, they are equivalent in order of magnitude, specifically due to the following theorems:*
  - $\mathbf{AM}[k] \subseteq \mathbf{IP}[k]$*, trivially.*
  - $\mathbf{IP}[k] \subseteq \mathbf{AM}[k+2]$.
  - $\mathbf{PSPACE} = \mathbf{IP}[\mathsf{poly}(n)] = \mathbf{MA}[\mathsf{poly}(n)]$.

**Theorem 3.** *Due to the strongly correlated nature of* **PSPACE** *and* **IP** *rooted from the motivation of their definitions, there should be at least some intuitions at this point that the following is true:*

$$\mathbf{IP} = \mathbf{PSPACE} \text{ [Sha92]}.$$

**Definition 12 (QIP).** *We give* **IP** *the additional power:*

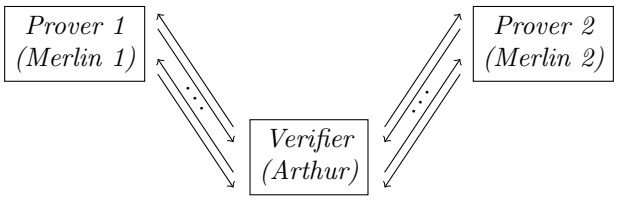
- *The verifier can be a* **BQP** *verifier.*
- *The messages sent can be quantum.*

**Theorem 4.** *It was actually shown that, at this point, quantum doesn't give any extra power (which makes the next result all the more surprising):*

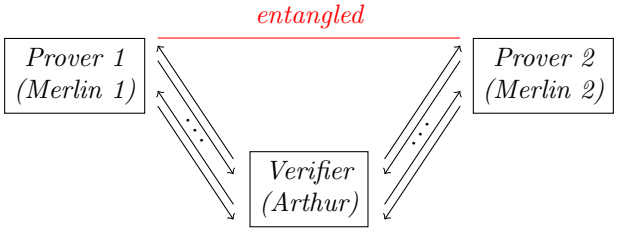
$$\mathbf{IP} = \mathbf{PSPACE} = \mathbf{QIP} \text{ [JJUW11]}.$$

# 3 $\underset{\mathbf{QIP}}{\mathbf{IP}} \to \mathbf{MIP} \to \mathbf{MIP}^*$

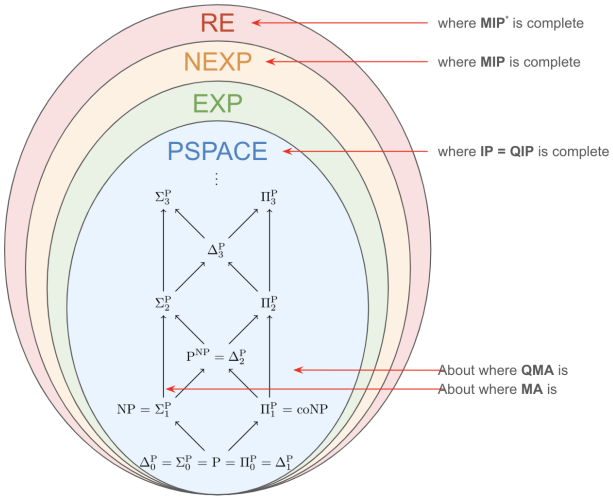
**Definition 13.** *The same as* **IP** *except that there can now be multiple provers:*




**Definition 14 ($\mathbf{MIP}^*$).** *The same as* **MIP** *except that there can be quantum entanglements between provers:*



# 4 Summary


RE
NEXP
EXP
PSPACE
where MIP* is complete
where MIP is complete
where IP = QIP is complete
About where QMA is
About where MA is
$\Sigma_3^P$
$\Pi_3^P$
$\Delta_3^P$
$\Sigma_2^P$
$\Pi_2^P$
$P^{NP} = \Delta_2^P$
$NP = \Sigma_1^P$
$\Pi_1^P = coNP$
$\Delta_0^P = \Sigma_0^P = P = \Pi_0^P = \Delta_1^P$

# Cryptography

# 1 Motivation

- Algorithm: Try to prove something is easy.
- Complexity: Try to prove something is hard. For this, one of my personal favorites:

  > "Go to the roots of calculations! Group the operations. Classify them according to their complexities rather than their appearances! This, I believe, is the mission of future mathematicians." — Evariste Galois

- Cryptography: Try to prove things are hard based on assumptions (which we call cryptographic primitives). Impagliazzo's five worlds:
  - Algorithmica: $\mathbf{P} = \mathbf{NP}$.
  - Heuristica: $\mathbf{P} \neq \mathbf{NP}$ but $\mathbf{NP}$ is easy on average.
  - Pessiland: $\mathbf{NP}$ is hard on average but OWFs don't exist.
  - Minicrypt: OWF exists but public key encryptions don't exist (PKE is also known as the asymmetric encryption).
  - Cryptomania: PKE exists.
  - (Obfustopia: Indistinguishability Obfuscator ($i\mathcal{O}$) exists.)

In particular, there are two fairly recent extensions of classical cryptography to quantum cryptography that I want to talk about.

# 2 Zero-Knowledge Proofs

## 2.1 Traditional Proofs

Here is an example, a traditional proof with $A$ and $A \implies B$ as axioms, then $B$ can be proved in the following steps:

- $A$ (axiom).
- $A \implies B$ (axiom).
- $B$ (rule of inference).

For a Prover to convince a Verifier is for Prover to come up with a proof that can be efficiently written down in **PSPACE** and efficiently checked for correctness in **P**. By what we said in the **IP** notes, if Verifier is deterministic, the class of proofs that Verifier can be convinced this way is **NP**. Then, **IP** is when Verifier has probabilistic power and the coin flips are private to the Verifier.

**Proposition 5** (Power Gap)**.** *There's no traditional proof for* Graph-Non-Isomorphism *but there is an* **IP** *for it (as the problem is in* **PSPACE** *which is equivalent to* **IP***).*

**Proposition 6** (Traditional proofs leave no room for privacy!)**.** *Think about this: If* Prover *is able to prove* Verifier *with a series of axioms and rules, then now* Verifier *can turn around and convince some third-party* $\text{Verifier}_2$ *by doing the same thing. That means,* Verifier *now has additional power that it used to not have before seeing the full proof!*

**Example 2** (Traditional Proof for Graph-Isomorphism)**.** Prover *shows* Verifier *the permutation.*

**Example 3** (**IP** for Graph-Isomorphism [**?**])**.** *Set-up: both* Prover *and* Verifier *know a pair of graphs* $(G_0, G_1)$ *and* Prover *knows permutation* $\phi$ *such that* $\phi(G_0) = G$*. The point is* Prover *can convince* Verifier *without showing the* Verifier *what the permutation is.*

1. Prover *randomly generates a permutation* $\pi$ *and computes* $C = \pi(G_1)$ *and sends* $C$ *to* Verifier*.*
2. Verifier *randomly picks a bit* $b$ *and sends* $b$ *to* Prover*.*
3. *If* $b = 0$*, then* Prover *sends* $\sigma = \pi \circ \phi$ *to* Verifier *(so* $\sigma(G_0) = \pi(G_1) = C$*). Or, if* $b = 1$*, then* Prover *sends* $\sigma = \pi$ *to* Verifier *(so* $\sigma(G_1) = C$*). That is,* Verifier *must pick the right* $\sigma$ *to put through for* $\sigma(G_b) = C$*.*
4. *Finally,* Verifier *accepts iff* $\sigma(G_b) = C$*.*

Keynote: Note that $\pi$ is picked uniformly randomly, so it conveys no additional information. Then, all Verifier has ever seen is $\pi \circ \phi$, so it also won't be able to gain additional information about $\phi$. Next, we formulate and quantify exactly what this difference is (i.e. the privacy preserving power of **IP** which is extra to Traditional-Proof systems).

## 2.2 The Notion of ZK

### 2.2.1 Background

**Asymmetry:**

Asymmetry is a concept embedded into the definition of **ZK**, and all it means is that the **ZK** privacy requirement concerns only with the positive case, i.e. where a proof (as in contrast with a disproof) is demanded. Here is why this is natural to consider:

- Think about example 3, we only specify what to do if $G_0 \cong G_1$ is indeed true, in which case Prover does need to convince Verifier. In a sense, one should think that there is only a job to do when the Prover does need to convince the Verifier and the privacy requirement is about leaking no information when Prover is doing this job for Verifier.
- On the other hand, when $G_0 \ncong G_1$ a trivial thing to do is just for Prover to abort, so that, again, all that Verifier still learns nothing about the proof other than that the isomorphism may be false. This is another way to look at why **ZK** about the negative case in this scenario isn't important to specify.

**Knowledge Act:**

What is knowledge? Or, at least, what we consider as knowledge when we say we want to leak no knowledge?

**Definition 15** (Knowledge Act)**.** *The following come for free and do not count as knowledge:*

- *Randomness.*
- *Computation that can be done in poly-time.*

*In summary, anything that can be computed by a* PPT *is free in terms of knowledge; otherwise, it is not free.*

**Remark 5** (Information vs. Knowledge)**.** *Here is a good place to differentiate between information and knowledge. For example, a random bit string contains a lot of information, but it nonetheless contains no knowledge.*

### 2.2.2 Interactively Defining ZK

**How to View ZK in IP?**

**Definition 16** ($((P,V)[x])$)**.** *Let $(P,V)[x]$ denotes the set of all possible sequences of messages sent by the* Prover*, $P$, in the current* **IP** *system. *Note: Why only what $P$ sent? This is because $V$ is capable of what it sends, so only what $P$ sends can possibly count towards knowledge.*

**Definition 17** ($\mathsf{VIEW}_{P,V}[x]$, Informal, Bad, Just For Now)**.** *$\mathsf{VIEW}_{P,V}[x]$, the set of a sequence of interactions between $P$ and $V$, can be seen as a distribution over $(P,V)[x]$.*

**Proposition 7. ZK** *for an* **IP** *system $(P,V)[x] \iff \mathsf{VIEW}_{P,V}[x]$ can easily be generated by $V \iff \mathsf{VIEW}_{P,V}[x]$ can easily be generated by* PPT*.*

**Subconscious / Simulator:**

To show that the **IP** we had for Graph-Isomorphism was indeed **ZK**, we show that the Verifier can single-handedly simulate the conversations generated by $(P,V)[G_0,G_1]$. In this case, we say that the Verifier is in the subconscious state:

**Definition 18** ($S_{\mathsf{PPT}}$, Verifier in the subconscious state)**.** *$S_{\mathsf{PPT}}$ is* Verifier *when* Prover *is "killed" and the* Verifier *is single-handedly trying to construct the conversation it would have had with the* Prover *with its* PPT *power.*

**Definition 19** (Original Conversation vs. Simulated Conversation)**.** *We call the conversation that $V$ and $P$ would have had the "original conversation" and the conversation that $V$ simulates single-handedly the "simulated conversation."*

**Example 4.** *A simulated conversation for* Graph-Isomorphism*:*

- Verifier *randomly chooses a permutation $\pi$.*
- Verifier *randomly chooses a permutation $b \in \{0,1\}$.*
- *Then,* Verifier *defines $C = \pi(G_b)$.*

*It is clear that, if we assume $G_0 \cong G_1$, then $C$ should be isomorphic to both of them.*

**Remark 6.** *Note the following about example 4:*

- *The conversation simulated and the original conversation are exactly the same other than the order (which doesn't matter because the two randomness are used independently). The idea is that $\phi$ is intrinsic to $G_0$ and $G_1$, so we don't need the $\phi$ that is the morph between the two graphs in order to generate the conversation.*
- *One gap is that, in the simulated conversation, $\pi$ is chosen randomly, but $\pi$ in the original conversation may be chosen with some other distribution (such as lexicographically the first one). So, we can just update the protocol and let the* Prover *also choose $\pi$ randomly.*

**Remark 7.** *Though example 4 is a one-round* **IP** *system, but this is not necessarily the case, as we could as well generalize this process to $k$-round.*

**Proposition 8.** *In this case, where the simulated conversation and the original conversation are practically the same, we say that they are indistinguishable (though how indistinguishable they are needs to be formally defined and will lead to different kinds of* **ZK** *proof [computational, statistical, information-theoretic]).*

**First Attempt of Formal Definition of ZK Property:**

**Definition 20** (**ZK** property, First Attempt)**.** *An* **IP** *system that decides $L$, denoted $(P,V)[x], \forall x \in L$ is said to have* **ZK** *property if $\exists S_{\mathsf{PPT}}$ s.t. $\forall x \in L$, $\mathsf{VIEW}_{P,V}[x] = S_{\mathsf{PPT}}[x]$.*

**Second Attempt of Formal Definition of ZK Property:**

The problem of the previous definition 20 is that it doesn't concern the possibility of $V$ being dishonest. That is, all that it asks for is just that the indistinguishability result applies to only one specific verifier $V$. What we could do instead is asking that this indistinguishability holds for all $V$:

**Definition 21** (**ZK** property)**.** *An* **IP** *system that decides $L$, denoted $(P,V)[x], \forall x \in L$ is said to have* **ZK** *property if $\forall V', \exists S_{\mathsf{PPT}}$ s.t. $\forall x \in L$, $\mathsf{VIEW}_{P,V'}[x] = S_{\mathsf{PPT}}[x]$.*

**Remark 8.** *Notice that in this definition we have two different* Verifiers*, one is the abstract $V$ that defines the set of all conversations that the* **IP** *protocol would be able to generate, and the other is the specific $V'$ that has a view and is able to generate the entire conversation single-handedly.*

**Remark 9.** *Notice how the* **ZK** *property is yet another asymmetric requirement that resembles soundedness. While soundedness requires that no dishonest* Prover *could inadvertently convince the* Verifier *of something that is not supposed to be true. Analogously, the* **ZK** *property is to ensure that no dishonest* Verifier *could learn more than zero-knowledge from any possible conversation.*

**Remark 10.** *The second version of the* **ZK** *property as given in definition 21 can be achieved. In fact, it can often be achieved by just some* **IP** *that satisfies the first definition 20.*

## 2.3 Proving That an IP System Satisfies ZK Property

### 2.3.1 What Does an IP System with ZK Property Look like?

**Remark 11.** *Because the* Verifier *needs to have enough power to simulate the messages that* Prover *sends it, so it must be the case that all the messages that* Prover *sends must be* PPT *computable based on what* Verifier *knows.*

### 2.3.2 Black-Box Simulation

In our section definition 21, we formulated it such that **ZK** property should hold for general verifiers, $V'$. So, in our security analysis, the internal workings of $V'$ shouldn't matter too much, which is why we introduce the black-box simulation version of the definition:

**Definition 22.** *We let $S_{\mathsf{PPT}}$ only have black-box access to $V'$ instead (so basically $V'$ acts as an oracle). Then, the definition, similarly to definition 21, is*

$$\exists S_{\mathsf{PPT}}, \forall V', \forall x \in L, \mathsf{VIEW}_{P,V'}[x] = S^{V'}[x].$$

**Remark 12.** *Definition 22 is actually a stronger definition than 21, because it is possible for $S_{\mathsf{PPT}}$ to know the internal working of $V'$, whereas, in this version of the definition, $S_{\mathsf{PPT}}$ has no information about $V'$ other than its I/O. Furthermore, this $\forall S_{\mathsf{PPT}}$ comes before $V$, which means that this single simulator works against all* Verifiers*.*

**Remark 13.** *For a typical definition of* **ZK** *property, definition* 21 *suffices and is standard. Definition 22 is for the purpose of showing that* **IP** *system is has* **ZK** *property.*

So, here is a quick summary of what we have so far:

- A zero-knowledge proof system (**ZKPS**) for $L$ is one such that all of the following are satisfied:
  - (Completeness) If $x \in L$, $\Pr[(P,V)[x] = 1] \geq 1 - \mathsf{neg}$.

- (Soundedness) If $x \notin L$, $\Pr[(P,V)[x]=1] \leq \frac{1}{2}$.
- (Zero Knowledge) $\forall V', \exists S_{\mathsf{PPT}}, \forall x \in L, \mathsf{VIEW}_{P,V'}[x] = S_{\mathsf{PPT}}[x]$.

- A stronger zero-knowledge proof system (**ZKPS**) for $L$, which we use to prove **IP** has zero-knowledge property, is one such that all of the following are satisfied:
  - (Completeness) If $x \in L$, $\Pr[(P,V)[x]=1] \geq 1 - \mathsf{neg}$.
  - (Soundedness) If $x \notin L$, $\Pr[(P,V)[x]=1] \leq \frac{1}{2}$.
  - (Zero Knowledge) $\exists S_{\mathsf{PPT}}, \forall V', \forall x \in L, \mathsf{VIEW}_{P,V'}[x] = S^{V'}_{\mathsf{PPT}}[x]$.

#### 2.3.3 Example: How Does $S_{\mathsf{PPT}}$ Produce a Conversation Indistinguishable from $\mathsf{VIEW}_{P,V'}$ with only oracle access to $V'$

The idea is that $S_{\mathsf{PPT}}$ keeps trying until $V'$ outputs something that signals it that it is the conversation that $V'$ would have induced. This way, the same single $S_{\mathsf{PPT}}$ could simulate any $V'$. Here are the steps:

1. $S_{\mathsf{PPT}}$ randomly chooses $b_i \in \{0,1\}$ and permutation $\pi_i$ during the $i$-th round. Then, $C_i = \pi_i(G_{b_i})$.
2. $S_{\mathsf{PPT}}$ sends $C_i$ to $V'$, and $V'$ outputs $d_i$.
3. $S_{\mathsf{PPT}}$ keeps the $i$-th round if $b_i = d_i$; keeps going without keeping it if not.

$*$After finishing to enumerate every possible guesses, $S_{\mathsf{PPT}}$ should be able to indistinguishably construct a $k$-round **IP** conversation.

#### 2.3.4 Some Nuances

**Remark 14.** *Here is a problem: Since we are requerying $V'$ many times, so every time we query $V'$ it can set $V'$ in a different state. But, we won't know anything about it though, since we know nothing about the inner workings of the oracle, nor can we reset its states.*

**Definition 23** (Rewinding)**.** *There are three types of rewinding:*

- *(Code) $S_{\mathsf{PPT}}$ has a copy of $V'$'s code, then $S_{\mathsf{PPT}}$ can simply restart the code.*
- *(Black Box) $S_{\mathsf{PPT}}$ cannot see the inner workings of $V'$ but have access to its random tape, then it can restart the* TM*.*
- *(Clones) If $S_{\mathsf{PPT}}$ has access to absolutely nothing of $V'$ but its I/O, it can try to clone several copies of $V'$.*

Corresponding to different types of access, we have the following **ZKPS** settings:

- (Full Power) $\forall V', \exists S_{\mathsf{PPT}}, S_{\mathsf{PPT}}[x] = \mathsf{VIEW}_{P,V'}[x]$. $S_{\mathsf{PPT}}$ works non-uniformly based on which $V'$ it's dealing with.
- (Code Power) $\exists S_{\mathsf{PPT}}, \forall V', \forall x \in L, S_{\mathsf{PPT}}(\mathsf{code}(V'), x) = \mathsf{VIEW}_{P,V'}[x]$. That is, a single $S_{\mathsf{PPT}}$ is powerful enough to simulate a **ZKPS** when it has access to $V'$'s code.
- (Black Box Power) $\exists S_{\mathsf{PPT}}, \forall V', \forall x \in L, S^{V'}_{\mathsf{PPT}}[x] = \mathsf{VIEW}_{P,V'}[x]$. That is, a single simulator is powerful enough to simulate any $V'$, even when it doesn't know $V'$'s inner workings.

**Remark 15.** *The above definitions are all listed in the order from the least restricted to the most in terms of what $S_{\mathsf{PPT}}$ has access to (the most restricted model would be a B.B. and the common definition for* **ZKPS** *following how it is naturally intended uses full power; the less restricted the weaker the definition).*

**Remark 16.** *What if $V'$ is intrinsically random? That is, what if, with the same input, $V'$ still outputs different things? Then the B.B. model would not work.*

## 2.4 But In Quantum?

There are several key differences in quantum that we must consider as compared to the classical case:

- Bad:
  - Some classically hard problems are broken: such as factoring by Shor's and certain lattice problems in the last decade. So, in a sense, the same cryptographic primitives in the classial world are harder to achieve in the quantum world.
  - Classical security analysis would fail, as there are additional quantum attacks available, and the quantum adversaries are hard to analyze (as one can see from the paper I linked and the future talk Francesco will give).
- Good:
  - Quantum protocols outperform classical protocols. For example, we know that quantum key distribution is information theoretically secure which is something known to be unattainable for classical key distribution.
  - There are additional crypto tools for quantum tasks: Encrypt quantum data.

## 2.5 Here is a Summary of Chain of Results Generalizing from Classical ZK Proof Systems to Quantum ZK Proof Systems

### 2.5.1 NP → QMA

**Theorem 5** ([GMW91])**.** *Every problem in* **NP** *has* **ZK** *proof system.*

What are the gaps to extend this to quantum?

- Rewinding techniques fail entirely in quantum, until new techniques were found by Wastrous [Wat06]. This new technique will make the protocol in [GMW91] still quantum-secure. However, this only applies to restricted cases.
- Quantum **ZK** were understood up to statistical, but [GMW91] concerns computational.

*Proof (Overview).* The idea is to reduce **ZK** for **QMA** to a problem that we know how to solve, which is **ZK** for **NP**. Firstly, we look at how to reduce from **ZK** for **NP** to **ZK** for **NP** through homomorphic encryption and fix the gap between **NP** and **QMA** in this process.

**Definition 24** (Homomorphic Encryption)**.** *Here's the set-up: let* $(pk, sk) \leftarrow \mathsf{Gen}(1^\lambda), ct \leftarrow \mathsf{Enc}(pk, m; r)$*, and* $m = \mathsf{Dec}(sk, ct)$*.* HE *is when the following are satisfied:*

- *(Secrecy): Let* $m, m'$ *be two messages,* $ct \leftarrow \mathsf{Enc}(pk, m; r)$ *and* $ct' \leftarrow \mathsf{Enc}(pk, m'; r)$ *should be indistinguishable.*

$$\begin{array}{ccc} m & & m' \\ \downarrow & & \downarrow \\ \mathsf{Enc}(pk, m; r) & \approx_\epsilon & \mathsf{Enc}(pk, m'; r) \end{array}$$

- *(Homomorphic) Similar to the homomorphic property in algebra and the functionality focuses on its verifiability.*

$$\begin{array}{ccccc} m & \longrightarrow & \boxed{f} & \longrightarrow & f(m) \\ \mathsf{Enc}(pk, m; r) \downarrow & & & & \uparrow \mathsf{Dec}(sk, \hat{f}\,[\mathsf{Enc}(pk, m; r)]) \\ \mathsf{Enc}(pk, m; r) & \longrightarrow & \boxed{\hat{f}} & \longrightarrow & \hat{f}\,[\mathsf{Enc}(pk, m; r)] \end{array}$$

With HE available, the idea is to have Verifier homomorphically evaluates Verification circuit on encrypted witnesses, just like the classical case, with the following gap dealt with in [BJSW16]:

- Need the right tools in quantum setting, like encoding by qubits.
- Need authentication to protect against dishonest Verifier with quantum power.

■

**Corollary 1.** *With this result that aligns* **QMA** *with* **NP** *in terms of* **ZK**, *notice that* **QIP** *is a generalization of* **QMA** *and* **IP** *a generalization of* **NP**. *Also, notice that* **PSPACE** = **IP** = **QIP**, *a lot of what one can say about* **IP** *being a* **ZK** *proof system can be similarly said about* **QIP**.

# 3 Non-Local Games → Compiled Games

## 3.1 Review

These are all from chapter 6 of our main source. In particular, reviewing 6.2 quantum correlations, 6.3 the **CHSH Inequality**, and 6.4 Bell's Theorem via **CHSH** would be helpful!

**Definition 25** (Nonlocal)**.** *Recall the locality requirement we have seen:*

> *Alice's choice of measurements (choosing between $A_1$ and $A_2$) does not affect the outcomes of Bob's measurement, and vice versa.*

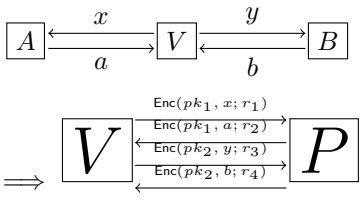
**Proposition 9.** *To classically test 2 quantum devices nonlocally, we use a* **Bell test** *like the* **CHSH game** *[it is known this this techniques can be extended to verify all* **BQP**, *by works in the last decade].*

> *Nonlocal correlations are "classical leash" on quantum systems.*

## 3.2 Current Landscape about This?

Here comes the big question of compiled games: how do we classically test 1 quantum device? We simulate locality using cryptography, in particular, a classical leash on a computationally bounded quantum device using the Learning-With-Error problem [BCM+21, Mah18].
Here is a line of current works:

- **MIP**$^*$ model is shown to be able to win certain "nonlocal games" (two or more players (Provers) who are only allowed to communicate with the 1 referee (Verifier)) [Bel64].
- Through a later line of research, this set-up where noncommunicating provers of the **MIP** model are allowed to share quantum entanglement has become one of the best understood omodels in quantum complexity theory (in the last 20 years).
- [KLVY23] very recently proposed the following transformation from multi-provers to a single prover, roughly:




  and showed that several **CHSH Game** properties were preserved.
- [NZ23] Better the analysis and showed more quantum properties.

# References


[BCM+21] Zvika Brakerski, Paul Christiano, Urmila Mahadev, Umesh Vazirani, and Thomas Vidick. A cryptographic test of quantumness and certifiable randomness from a single quantum device. *Journal of the ACM (JACM)*, 68(5):1–47, 2021.

[Bel64] John S Bell. On the einstein podolsky rosen paradox. *Physics Physique Fizika*, 1(3):195, 1964.

[BJSW16] Anne Broadbent, Zhengfeng Ji, Fang Song, and John Watrous. Zero-knowledge proof systems for qma. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 31–40, 2016.

[dW23] Ronald de Wolf. Quantum computing: Lecture notes. 2023.

[GMW91] Oded Goldreich, Silvio Micali, and Avi Wigderson. Proofs that yield nothing but their validity or all languages in np have zero-knowledge proof systems. *J. ACM*, 38(3):690–728, jul 1991.

[GS86] Shafi Goldwasser and Michael Sipser. Private coins versus public coins in interactive proof systems. In *Proceedings of the eighteenth annual ACM symposium on Theory of computing*, pages 59–68, 1986.

[JJUW11] Rahul Jain, Zhengfeng Ji, Sarvagya Upadhyay, and John Watrous. Qip = pspace. *J. ACM*, 58(6), dec 2011.

[KLVY23] Yael Kalai, Alex Lombardi, Vinod Vaikuntanathan, and Lisa Yang. Quantum advantage from any non-local game. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, pages 1617–1628, 2023.

[Mah18] Urmila Mahadev. Classical verification of quantum computations. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 259–267, 2018.

[MW05] Chris Marriott and John Watrous. Quantum arthur–merlin games. *computational complexity*, 14(2):122–152, 2005.

[NZ23] Anand Natarajan and Tina Zhang. Bounding the quantum value of compiled nonlocal games: from chsh to bqp verification. In *2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1342–1348. IEEE, 2023.

[Sha92] Adi Shamir. Ip = pspace. *J. ACM*, 39(4):869–877, oct 1992.

[Wat06] John Watrous. Zero-knowledge against quantum attacks. In *Proceedings of the Thirty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '06, page 296–305, New York, NY, USA, 2006. Association for Computing Machinery.