Introduction to Quantum Interactive Proof Systems and Applications

Liam Salt - APMA 990

April 7, 2022

Abstract

In this paper, we introduce quantum interactive proof systems, demonstrate their advantages over classical systems, and showcase their utility by using them to prove certain properties of finite groups. We show that any language in PSPACE has a 2-round quantum interactive proof system, which is strictly more powerful than the classical case. Finally, we consider quantum interactive proof systems in the context of group theory via the group non-membership problem. We show that succint proofs for non-membership are possible with a quantum proof system, which is impossible classically. Construction of efficient proofs for group non-membership also allows us to approach other group theoretical problems, such as: finding the maximal normal subgroup, and whether an integer N divides the order of a group.

1 Introduction

1.1 Some Computer Science Background

Before we can begin to define a quantum proof system, we first must introduce some classical computing formalism and terminology. In classical computing, an *interactive proof system* is a Turing machine that encapsulates the mathematical idea of a proof. It models a proof as a sequence of communications between two actors: a potentially dishonest *prover*, and a skeptical, honest *verifier*. Through various *rounds* of communication, the prover attempts to convince the verifier to accept their proof. The default convention is to assume the verifier is constrained to be a deterministic polynomial-time TM, whereas we make no constraints to the prover's computational power. Additionally, an interactive proof system is assumed to satisfy both *completeness* and *soundess*, defined below:

Definition. *An interactive proof system is said to be complete or to satisfy completeness if for any true statement, the prover can always convince the verifier of its validity.*

Definition. *An interactive proof system is said to be sound if for any false statement, the prover can only with negligible probability convince the verifier of its validity.*

Remark. *Often, many of these conditions are loosened in various ways, e.g. the prover only needing to convince the verifier up to some probability.*

Now we can more formally define a (deterministic) interactive proof system.

We say that a formal language L has a deterministic k-round interactive proof system with error probability ε with prover verifier pair (P, V) if all of the following:

1. (P, V) either accepts or rejects any input after k rounds
2. $\forall x \in L, (P, V)$ accepts x with probability 1
3. $\forall x \notin L, (P, V)$ accepts x with probability ε

Given this definition, a familiar, degenerate example is to show that any language in **NP** has a $1-$round interactive proof system with error probability 0. We recall that one definition of **NP** is as the set of decision problems whose positive answers can be verified in polynomial time by a deterministic Turing machine. Accordingly, if a problem is in **NP**, then by definition on input our prover can produce a polynomial-sized certificate, which can be verified in polynomial-time exactly when the input is in the language. In fact, the class of languages which have a deterministic interactive proof system, termed **dIP**, is exactly equal to **NP**. We can think of this as meaning that we can capture the full power of deterministic interactive proof systems with the class **NP**.

We can generalize the above definition by allowing the verifier to also have access to a random number generator, with which it may generate and make use of random bits in each round. The addition of the verifier's access to randomness brings us to the full definition of an interactive proof system. The complexity class of all languages that have interactive proof systems is termed **IP**, and it can be shown [potentially in this paper] that **IP**=**PSPACE**. In other words, problems which can be accepted by an interactive proof system are exactly those problems which require only polynomial-sized space (memory) to answer.

1.2 Arthur-Merlin Protocol

Related to **IP**, is another class which will become important in the quantum case is **MA**, which stands for Merlin-Arthur (of King Arthur fame), and is comprised of languages which have a particular type of interactive proof system known as an Arthur-Merlin protocol. This class was originally described in [Bab85] and [Gol89].

Definition. *A language L is said to have an Arthur-Merlin protocol with prover-verifier pair (Merlin, Arthur) if:*

1. *Arthur has access to random bits $y \in B_m$, which are public (known to Merlin)*
2. *If $x \in L$, then $\exists z \in B_n$ such that $P(\text{Arthur accepts } x|y, z) \geq \frac{2}{3}$.*
3. *If $x \notin L$, then $\forall z \in B_n$ such that $P(\text{Arthur accepts } x|y, z) \leq \frac{1}{3}$.*
4. *n, m are both polynomial-length*

Equivalently, we say $L \in \mathbf{MA}$. For clarity, in the above definition z is the polynomial-sized proof certificate provided by Merlin, and whether Arthur accepts Merlin's proof depends on both the string sent by Merlin and the random string produced by Arthur.

The probability conditions given above are quite flexible. Given a language meeting the definition above, it can be shown that we can induce a new Arthur-Merlin pair for that language that achieves perfect completeness, and soundness with error probability at most $\frac{1}{2}$.

It will be important to see how **MA** relates to other complexity classes:

Proposition. ***MA*** *contains both* ***NP=dIP*** *and* ***BPP****.*

Proof. The second containment is easy to see given the definition of **BPP**, as Arthur has access to random bits and a polynomial-time Turing machine (with acceptance prob $\frac{2}{3}$ as required in **BPP**); therefore, Arthur doesn't need to send anything to Merlin, as he is capable of solving any problem in **BPP** himself.
Similarly for **NP**, Merlin can solve the problem then send Arthur the proof certificate which he can verify deterministically, i.e. without needing his random bits. □

2 Quantum Interactive Proof Systems

Now that we have seen the definition of interactive proof systems in the classical case, we can discuss what happens if we allow the Turing machines access to quantum computations. It has been observed that allowing access to quantum computation lends itself to significant speed-up in many classical problems, including: from integer factorization, computing discrete logarithms, efficient unstructured search, etc., so it makes sense to ask whether any of the above definitions in section 1 are significantly affected by access to quantum circuits.

Remark. *The Turing machine model is sufficient, but for the remainder of this paper we will use the equivalent circuit model of quantum computation.*

Specifically, a quantum interactive proof system is one where the verifier has access to quantum circuits which can solve problems in **BQP**, while the prover is still unrestricted (within the boundaries of quantum mechanics). The convention is for the messages exchanged between prover and verifier to be encoded as quantum states in the usual $n-$qubit $z-$basis. Some relevant terminology is that we call the class of all languages that have quantum interactive proof systems Quantum **IP** or **QIP**.

Definition. *A language L is said to have a quantum interactive proof system with error probability ε, or to be in* ***QIP****, if:*

1. *The verifier accepts or rejects any input x using a quantum circuit solving problems in* ***BQP****.*
2. *If $x \in L$, then given the polynomial-many actions of the prover, the verifier accepts x with probability* 1*.*

3. *If $x \notin L$, then regardless of the actions of the prover, the verifier accepts x with probability ε.*

We can see that this definition for **QIP** looks quite similar to the classical definition of **IP**, which we discuss more in the next section. In its current state **QIP** is actually only as powerful as **IP**, and in fact, **QIP**=**IP**=**PSPACE**.

2.1 Relationship between QIP and PSPACE

Despite the simplicity of the claim at the end of the previous section, we can say a bit more about the relationship between **QIP** and **PSPACE**. Namely, we can put a very nice bound on the number of queries required in a quantum interactive proof system to determine any language in **PSPACE**. We introduce the first theorem, which is the principal subject of [Wat99]:

Theorem. *Every language in **PSPACE** has a 2−round quantum interactive proof system with exponentially small error probability.*

In other words, we can prove that **PSPACE** $\subseteq$ **QIP[2]**, the class of languages with 2−round quantum interactive proof systems, a much tighter bound than what was claimed in the previous section. Importantly, this also shows that in the constant round case, **IP[k]** is strictly outperformed by **QIP[2]**, demonstrating the power of moving to quantum computations.

To prove the above theorem, we need to show that any language in **PSPACE** has such a proof system, but it will be sufficient to prove the theorem for a language which is **PSPACE**-complete. In particular, we show the theorem for the language of true quantified boolean formulas, that is the language of true expressions $Q_1 x_1 \cdots Q_n x_n f(x_1, \ldots, x_n)$, where Q_i is a universal or existential quantifier $(\forall, \exists)$, x_i are boolean valued variables, and f is a boolean formula $B_n \to B$ consisting of operations $\wedge, \vee, \neg$,etc on the variables $x_1, \ldots, x_n$.

There is a classical protocol outlining an interactive proof system for the QBF problem [She92], but this is general takes $N = \binom{n+1}{2} + n$ rounds of communication. A full description of this classical protocol is not shown in this paper, but it is outlined in [Wat99]:
A general outline of the algorithm for a formula Q is:

1. The verifier sends the prover a random element r_1 of a finite field, and gets a corresponding function f_1 in response from the prover.

2. They continue this exchange N times, with each response polynomial f_i being at most $\deg d$ and depending only on the previously received random elements $r_1, \ldots, r_i$.

3. After receving the final f_N, the verifier inputs all of the data they have collected into a single predicate: $E(Q, r_1, \ldots, r_N, f_1, \ldots, f_N)$

4. The verifier accepts if and only if $E(Q, r_1, \ldots, r_N, f_1, \ldots, f_N)$ evaluates to true.

After some analysis, this serves to prove that in the classical case, **IP**=**PSPACE**, and we therefore have already that **PSPACE**⊆**QIP**, as we can simulate the classical circuit in the quantum case. Thus, it remains to show that allowing our system access to quantum circuits helps to reduce the number of rounds.

Now, we outline a 2−round quantum protocol for the language of true quantified boolean formulas.

We construct a quantum circuit which consists of three collections of registers: $\mathbf{R}_{i,j}, \mathbf{S}_{i,j}$, and $\mathbf{F}_{i,j}$, for $1 \leq i \leq m$ and $1 \leq j \leq N$, where $N = \binom{n+1}{2} + n$ as above, and $m = q(n)$ for some polynomial q. The mN registers found in the collections $\mathbf{R}$ and $\mathbf{S}$ each consist of k qubits, where $\mathbb{F}_{2^k}$ is some chosen finite field (In the classical protocol, a finite field is chosen such that it is as large as computationally practical, as the error rate goes down exponentially in k). Each register in $\mathbf{F}$ consists of $k(d+1)$ qubits, where d is the total length of the inputted formula. Each $F_{i,j}$ represents a polynomial of degree at most d with coefficients in $\mathbb{F}_{2^k}$, for example if $d = 3$ and $k = 2$, we represent the polynomial $3x^3 + 2x + 1$ as:

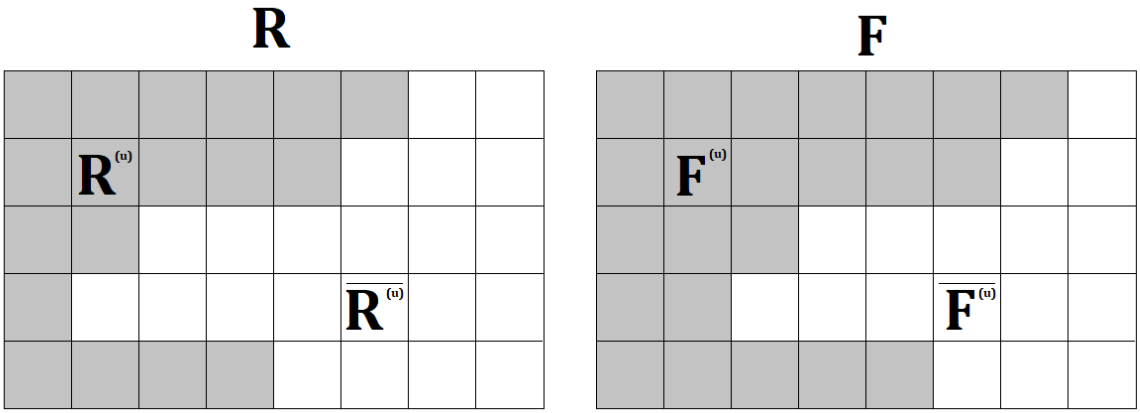
$$3x^3 + x + 1 \rightarrow |11\rangle|00\rangle|10\rangle|01\rangle$$

Additionally, the verifier needs to store classically the value of a vector $u \in \{1, \ldots, N\}^m$. According to the vector u, we partition the collections R and F into two parts:

$$R^{(u)} = \{R_{i,j} | j \in \{1, \ldots, u_i - 1\}\}, \overline{R^{(u)}} = R \setminus R^{(u)}$$

$$F^{(u)} = \{F_{i,j} | j \in \{1, \ldots, u_i\}\}, \overline{F^{(u)}} = F \setminus F^{(u)}$$

To summarize, we have defined three 2D arrays of registers, and partitioned two of them according to a vector. An example of $\mathbf{R}$ and $\mathbf{F}$ with $N = 8$, $m = 5$, and $u = (7, 6, 3, 2, 5)$ is shown below:



The protocol proceeds as follows: First, the prover sends $\mathbf{R}$, and $\mathbf{F}$ which they calculate using their unlimited computing power. Then, using the classical verification protocol described in [She92] for each pair of rows in the above table ($\mathbf{R}_i$,$\mathbf{F}_i$), the verifier rejects the proof if any pair for $i = 1, \ldots, m$ contain an invalid proof that the input formula evaluates

to true. The verifier chooses vector u uniformly randomly from $\{1,\ldots,N\}^m$ and sends it to the prover along with $\overline{\mathbf{F}^{(u)}}$. In the second round, the prover sends S to the verifier, and they calculate $\mathbf{R}_{i,j}$-$\mathbf{S}_{i,j}$ for each pair (i,j). Finally, the verifier applies $H^{\otimes k}$ to each register of $\overline{\mathbf{R}^{(u)}}$ and accepts if and only if $\overline{\mathbf{R}^{(u)}}$ contains only zeros.

Some notes: the entries in $\mathbf{R}$ and $\mathbf{F}$ correspond to the random values r_i and functions f_i from the classical protocol. A full proof of correctness is found in [Wat99], but we provide an outline here.

Proof. First suppose that a given formula Q is true. We want to show that the verifier accepts it with certainty. The prover also creates a random $m \times N$ matrix R of values in $\mathbb{F}_{2^k}$, where m, N, and k are defined as above. Using the values of R, the prover also generates a corresponding $m \times N$ matrix of correct polynomials $C(R)$. Each row $C(R)_i$ is formed by defining polynomials $f_{i,j}$ as in the classical protocol, where instead of getting the random values r_j from the verifier, they are taken from the corresponding row R_i of the generated matrix R. Thus, $C(R)_{i,j} = f_{i,j}(R_{i,1},\ldots,R_{i,N})$, for $i = 1,\ldots,m, j = 1,\ldots,N$. The prover then prepares the following state:

$$\frac{1}{\sqrt{2^{kmN}}} \sum_{i,j} |R_{i,j}\rangle |C(R)_{i,j}\rangle$$

Forming the registers $\mathbf{R}$ and $\mathbf{F}$ as in the above protocol $\mathbf{R}_{i,j}$ to $\mathbf{S}_{i,j}$. The prover then sends this state to the verifier. As we're assuming Q is true, the honest verifier will not reject at this point, and they will return $\overline{\mathbf{F}^{(u)}}$ and u to the verifer in response. Then, for each (i,j) define a unitary:

$$U_{i,j} : |R\rangle|0\rangle \mapsto |R\rangle|C(R)_{i,j}\rangle$$

The prover then applies $U_{i,j}^{-1}$ to registers $\mathbf{S}_{i,j}$ and $F_{i,j}$ for all $F_{i,j}$ in $\overline{\mathbf{F}^{(u)}}$, which keeps $\mathbf{S}$ the same, but returns all of $\overline{\mathbf{F}^{(u)}}$ to $|0\rangle$. After the verifier subtracts each $\mathbf{R}_{i,j}$ from each $\mathbf{S}_{i,j}$, the registers in $\overline{\mathbf{R}^{(u)}}$ will no longer be entangled with any other registers; therefore each register in $\overline{\mathbf{R}^{(u)}}$ is sent to the zero state, and the verifier will accept Q with certainty.
In the case where Q is not a true formula, the verifier will accept with some error probability. An upper bound for this probability is determined by m and k. If we choose $m = (d+1)N$ and $k = 2d + 6 + \lceil \log(dm \rceil)$, then $\varepsilon < 2^{-d}$, i.e. the error probability is exponentially small in the length of Q. □

Therefore, we have proved that **PSPACE⊆QIP[2]**; moreover, there is an $\varepsilon < 2^{-|x|}$ protocol for any language L in **PSPACE**.

2.2 QMA

Having now introduced quantum interactive proof systems and the classical class **MA**, we can discuss its natural extension **QMA**.

Definition. *A language L is in said to be in the class **QMA** if there exists a family of polynomial-time quantum circuits such that:*

1. *If $x \in L$, then there exists a state $|\Psi\rangle$ such that $P(\text{Circuit accepts } |\Psi\rangle) > \frac{2}{3}$*

2. *If $x \notin L$, then for any state $|\Psi\rangle$, $P(\text{Circuit accepts } |\Psi\rangle) < \frac{1}{3}$*

There are several important results to do with this class. For example, it has been shown by Kitaev that $\mathbf{QMA} \subseteq \mathbf{P}^{\sharp P}$[Kit99]. Another result of Marriott and Watrous is that $\mathbf{QMA} \subseteq \mathbf{PP}$[Mar05], and further, if the reverse containment holds and $\mathbf{QMA} = \mathbf{PP}$, it can be shown that $\mathbf{PH} \subseteq \mathbf{PP}$[Vya03].

3 Group Theoretic Applications

Having introduced it in the previous section, we proceed to discuss **QMA** in the context of group theory. The principal result of [Wat00] is a proof of the fact that the group non-membership problem is in **QMA**. This section will review the proof of this fact, along with some corollaries that allow us to recover various other group properties.

3.1 Group Non-Membership

We start by defining the group non-membership problem (GNP).

Definition. *Given a finite group G and $k+1$ elements $g_1, g_2, \ldots, g_k, h \in G$, the group non-membership problem asks whether h is in the subgroup generated by the elements g_i,i.e. is it the case that $h \in (g_1, \ldots, g_k) \subseteq G$.*

3.2 Other Applications/Open Problems

References

[Bab85] L. Babai, Trading Group Theory for Randomness, Proceedings of the Seventeenth Annual ACM Symposium on the Theory of Computing, 1985, 421-429.

[Gol89] S. Goldwasser, M. Sipser. Private coins versus public coins in interactive proof systems, Randomness and Computation, volume 5 of Advances in Computing Research, JAI Press, 1989, 73-90.

[Kit99] A. Kitaev, Quantum NP, Talk at AQIP'99: Second Workshop on Algorithms in Quantum Info

[Mar05] C. Marriott, J. Watrous, Quantum Arthur-Merlin Games, 2005. https://doi.org/10.48550/arXiv.cs/0506068

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

[Vya03] M. Vyalyi. QMA=PP implies that PP contains PH, 2003

[Wat99] J. Watrous, PSPACE has 2-round quantum interactive proof systems, 1999. https://doi.org/10.48550/arXiv.cs/9901015

[Wat00] J. Watrous, Succinct quantum proofs for properties of finite groups, 2000. https://doi.org/10.48550/arXiv.cs/0009002