University of Illinois, Urbana Champaign
ECE 498EC Quantum Information Theory

*Author:* Nishant Kumar
*Date:* December 20, 2020

**PROJECT**
**1**

# Zero-knowledge against quantum attacks[1]

In this report, we explore interactive proof systems which are zero-knowledge not only against classical polynomial-time adversaries, but also against quantum polynomial-time adversaries. Watrous [3] devised a technique for such a simulation which works in the quantum setting for many interactive proof systems. We look at the techniques behind these simulator constructions, by analyzing the zero-knowledge property for the GMW protocol for the Graph Isomorphism problem [1]. We begin with details on the problem setting and motivation and then move on to formalize the problem, followed by analyzing the zero-knowledge property for GMW protocol for the graph isomorphism problem.

## 1.1 Introduction and motivation

Interactive proofs are proof systems which are executed interactively between two parties - say the prover (**P**) and verifier (**V**). The two entities have a common statement and are interested in verifying if its true or not. The prover is assumed to have unlimited computational resources, but cannot be trusted, while the verifier, which can be trusted, is a computationally bounded machine and which cannot verify the statement on its own. In such a case, the prover is trying to convince the verifier of the validity of the statement by exchaning rounds of messages back and forth.

The notion of zero-knowledge for interactive proofs was introduced in the work of Goldwasser et al. in 1985 [2] and is of utmost importance in cryptography. In the above described setting of interactive proofs, it refers to the ability of the prover to convince the verifier of the validity of the statement without revealing anything more than statement's validity itself. The notion has been put to use in central feasibility results in theoretical cryptography.

Typically, the prover and verifier in such an interactive proof system are assumed to be classical entities. A long line of work has explored this setting and we know that all languages in NP have a zero-knowledge proof system in such a setting.

However, with quantum computing capabilities, the scenario changes. Ideally, we would still like to have the same desired properties for the proof system as in the classical world. While some desired properties like completeness (ability of an honest prover to convince an honest verifier of the validity of a true statement) and soundness (inability of a cheating prover to convince an honest verifier of the validity of a false statement) continue to hold, the property of zero-knowledge breaks down in some cases (or at least the same technique to prove that the proof system is zero-knowledge in the classical world no longer applies).

Lets try to intuitively understand the reason behind this. To define zero-knowledge, the notion of the verifier not learning anything beyond the validity of the statement needs to be formalized. This is

[1]https://arxiv.org/abs/quant-ph/0511020

done by requiring the existance of a polynomial-time simulator which given only the input statement can come up with a transcript of the protocol where the verifier accepts. But one might be puzzled by the following then: if the simulator can in some sense cause the verifier to accept without ever knowing the witness of the statement to be proven, why can't a real cheating prover follow the same strategy and convince the verifier of the validity of a statement for which it knows no proof? The reason is that the simulator is typically bestowed with additional powers - such as giving it black-box access to the code of the verifier. Given this black-box code of the verifier, a typical simulator in a proof of zero-knowledge in the classical world would work by feeding the verifier machine with parts of a randomly guessed transcript and observing its output. If the output is as expected then it can continue and produce an accepting transcript. If the output is not as expected, the simulator can "rewind" the verifier machine to an earlier state and rerun on different inputs. By gathering information from rerunning the verifier on different related inputs in this way, the simulator is able to produce an accepting transcript.

But things change in the quantum world. The central issue which prevents the same technique to work as is over here, is that in the quantum world since the verifier takes as input and output quantum state, the simulator can no longer trivially rewind the verifier to an earlier state. While in the classical world this rewinding is easy because the simulator can copy and keep states of the verifier (and rerun the verifier from those states), in the quantum world, the no cloning theorem prevents it from doing such a copying of the state of the verifier.

Given this state of affairs, it is no longer clear if a proof system which is proven zero-knowledge against classical polynomial-time verifiers is still zero-knowledge against quantum polynomial-time verifiers. It could very well be the case that while the proof system is still zero-knowledge in reality, its only the limitation of our current proof techniques. Or it could be the case that this means there are attacks on the zero-knowledge property which remain to be discovered.

In his work, Watrous [3] showed that many proof systems which are zero-knowledge in the classical world are also in fact zero-knowledge in the quantum world. He did this by demonstrating the existance of a quantum poly-time simulator in these settings, which works quite analogously to its classical counterpart, but in addition to doing some more steps. In this report, we explore in detail this simulator's construction and the corresponding proof that the system still remains zero-knowledge. We do this by looking at the simulator's construction for the GMW proof system for the graph isomorphism problem from 1991 [1].

## 1.2 Background and formalizing the problem statement

Lets now try and formalize the notion of zero-knowledge which we will be proving (definitions and parts here are used from [3]. )

An interactive proof system is specified by a pair $(\mathbf{P}, \mathbf{V})$ representing the honest prover and honest verifier strategies. The completeness property of such a system refers to interactions between $(\mathbf{P}, \mathbf{V})$, i.e. the ability of an honest prover to convince an honest verifier of the validity of a true statement. The soundness property refers to interactions between $(\mathbf{P}^*, \mathbf{V})$, where $\mathbf{P}^*$ refers to a prover strategy which arbitrarily deviates from the honest prover strategy. In other words, it refers to the inability of a cheating prover to convince an honest verifier of the validity of a false statement. The zero-knowledge property refers to interactions between $(\mathbf{P}, \mathbf{V}^*)$, where $\mathbf{V}^*$ refers to a verifier strategy which arbitrarily deviates from the honest verifier strategy. In other words, this refers to the inability of a cheating verifier to learn anything more than the validity of the statement itself from its interaction with an honest prover on a true statement.

Since in this report we are only concerned with the property of zero-knowledge, lets try and formalize this further. Lets do this for the classical setting first. The prover and verifier have as common input a promise problem - $A = (A_{yes}, A_{no})$. In addition, an arbitrary verifier $\mathbf{V}^*$ takes as input an auxiliary input string $w$, which is known only to the verifier and which may affect verifier's actions in the protocol. The verifier upon interaction with the prover produces a string as output.

The size of the auxiliary input string $w$ and the output string are given by polynomial functions - $n, m : \{0,1\}^* \to \mathbb{N}$, which are polynomially bounded functions. So, if the common input string is $x$, then the size of the auxiliary input is given by $n(x)$ and the size of the output is given by $m(x)$. Since randomness is typically involved in these interactive proofs, the verifier's output is denoted by the random variable - $(\mathbf{P}, \mathbf{V}(w))(x)$.

A simulator, $\mathbf{S}$, in this classical world is a probabilistic poly-time algorithm, which takes as input strings $w, x$ with $|w| = n(x)$ and produces an output string of length $m(x)$. The random variable representing the simulator's output is given by $\mathbf{S}(w, x)$. For a given promise problem $A$, an interactive proof system $(\mathbf{P}, \mathbf{V})$ is said to be zero-knowledge if for every verifier strategy $\mathbf{V}^*$, there exists a simulator $\mathbf{S}$, such that $(\mathbf{P}, \mathbf{V}(w))(x) \cong \mathbf{S}(w, x)$. The $\cong$ symbol used here refers to indistinguishability of the two distributions. Different versions this indistinguishability give rise to zero-knowledge proofs with different properties then. A zero-knowledge proof system is said to be perfect zero-knowledge if the simulator can report a failure with some small probability, but otherwise the two distributions are identical. It is said to be statistical zk, if the two distributions have only a negligible statistical difference. And finally it is said to be computational zk, if a probabilistic polynomial time distinguisher cannot distinguish the two distributions with better than negligible probability.

In the quantum case, since we are looking at the zero-knowledge property, the adversarial verifier is allowed to be quantum, while the honest prover is assumed to be classical. Zero-knowledge property in this case is defined similarly, except that the verifier in the real-interaction now takes quantum states as input and produces qunatum output, i.e. it takes some $n(x)$ qubits as inputs and produces $m(x)$ qubits as output, where $n, m$ are polynomially bounded functions. This interaction of $\mathbf{V}^*$ with $\mathbf{P}$ in the real-world produces a CPTP map $\Phi_x : L(\mathcal{W}) \to L(\mathcal{Z})$, where $L(\mathcal{W})$ refers to linear operators on the hilbert space of the input $n(x)$ qubits, while $L(\mathcal{Z})$ refers to the hilbert space of the output $m(x)$ qubits. Similarly, a simulator takes the auxiliary state as input in the form of $n(x)$ qubits and produces $m(x)$ qubits as outputs, and this produces a CPTP map $\Psi_x : L(\mathcal{W}) \to L(\mathcal{Z})$. To define indistinguishability in a similar manner as the classical world, we need some measure of the distance between the two CPTP maps - $\Phi_x$ and $\Psi_x$. However, in this report we will only be looking at perfect zero-knowledge, where the two maps will turn out to be identical.

## 1.3 The GMW Graph Isomorphism protocol

The Graph isomorphism problem (henceforth referred to as the GI problem) is defined as the following: given two undirected simple graphs $G_0, G_1$, one needs to tell whether the two graphs are isomorphic to each other. We say 2 graphs $G_0, G_1$ are isomorphic (written as $G_0 \cong G_1$) if there exists a mapping $\pi$ from vertices in $G_0$ to vertices in $G_1$ such that the edges are preserved under this mapping.

An interactive zero-knowledge proof for the GI problem was proposed back in 1991 by Goldreich, Micali and Wigderson [1]. The protocol can be shown to have perfect completeness, a soundness error of $\frac{1}{2}$ and perfect zero-knowledge against classical polynomial time verifiers. In this report, we aim to show that the protocol still has perfect zero-knowledge against poly-time quantum verifiers.

The protocol of [1] is shown formally in Figure 1.1. Completeness follows because $\tau(G_a) = \pi\sigma^a(G_a)$. But if $a = 0$, then this is $\pi(G_0) = H$ (by definition of how $H$ was defined). If $a = 1$, then this is $\pi(\sigma(G_1)) = \pi(G_0) = H$ again. Hence, completeness follows.

For soundness, note that a cheating prover $\mathbf{P}^*$ can do nothing better than guess what bit $\mathbf{V}$ will ask. Hence, soundness is $\frac{1}{2}$. Sequential composition of this protocol can be used to get the soundness down to negligible.

We now show the protocol is perfect zero-knowledge against classical poly-time verifiers. The simulator would work as follows: guess the bit which $\mathbf{V}$ would use - say $b$. Also choose a random permutation $\pi$. Prepare $H = \pi(G_b)$ and send $H$ over to $\mathbf{V}$. If $\mathbf{V}$ sends the same bit, then complete the simulation by sending $\pi$, else rewind $\mathbf{V}$ to start and start with fresh choice of $\pi$ and $b$. The simulator succeeds with overwhelming probability in polynomial number of steps.

Assume the common input to prover($\mathbf{P}$) and verifier($\mathbf{V}$) is a pair of simple, undirected graphs $G_0, G_1$ of $n$ vertices - $\{1, 2 \ldots n\}$.

1. Let $\sigma \in S_n$ be a permutation satisfying $\sigma(G_1) = G_0$ if $G_0 \cong G_1$, or the identity permutation otherwise. $\mathbf{P}$ chooses permutation $\pi \in S_n$ at random and sends over $H = \pi(G_0)$ to $\mathbf{V}$.

2. $\mathbf{V}$ chooses a bit $a \in \{0, 1\}$ uniformly at random and sends $a$ to $\mathbf{P}$.

3. $\mathbf{P}$ sends over $\tau = \pi\sigma^a$ to $\mathbf{V}$.

4. $\mathbf{V}$ accepts if $\tau(G_a) = H$.

FIGURE 1.1: The GMW protocol for the graph isomorphism problem [1]. $S_n$ refers to the set of all possible permutations.

**Proof that the GMW GI protocol is quantum zero-knowledge** We shall now see that it is indeed possible to construct a simulator which works against poly-time quantum verifiers in the GMW GI protocol. For ease of analysis, we consider $\mathbf{V}$ to be of the following form and show this is without loss of generality.

1. $\mathbf{V}$ takes as input (in addition to $G_0, G_1$) the quantum register $W$, representing the auxiliary input to the verifier. Also, it has two additional workspace registers - a poly-size quantum register $V$ and a single qubit register $A$. Both $V$ and $A$ are initialized to the all-zero state before the protocol starts.

2. $\mathbf{P}$ then sends its first message $H$ to $\mathbf{V}$. Assume this first message is stored in the quantum register $Y$ (we introduce a quantum register for this because we will anyway need this in our simulator's analysis). $\mathbf{V}$ applies the unitary transform $V_H$ to registers $W, V, A$. It then measures register $A$ wrt the standard basis to get the bit $a$ and sends over this bit $a$ to $\mathbf{P}$.

3. $\mathbf{P}$ responds with some $\tau \in S_n$. Assume this is stored in a quantum register $Z$. $\mathbf{V}$ outputs the complete state of the registers - $W, V, A, Y, Z$.

Notice that at the end of the protocol, instead of outputting an accept-reject decision, $\mathbf{V}$ outputs the complete state of the registers it has. This is without loss of generality because any general verifier $\mathbf{V}$ can be taken to do some post-processing on these registers and then output the final accept-reject decision. Since the same post-processing can be applied on the output of the simulator, we just look at the distribution prior to applying this post-processing, i.e. we look at the state of all the quantum registers $\mathbf{V}$ has.

Also, note that by our description above, $\mathbf{V}$ is completely specified by the collection of unitary operators - $V_H$ which is the only operation performed by $\mathbf{V}$ on its registers.

In the following analysis, $G_0 \cong G_1$ since we are analyzing the zero-knowledge property, wherein the statement is assumed to be true.

**Notation** Lets setup some notation before we move further. We shall use $\mathcal{B}$ to denote the hilbert-space associated with the register $B$. Similarly, for all the other registers we will be using. Also, $L(\mathcal{A}, \mathcal{B})$ denotes the linear map from hilbert-space $\mathcal{A}$ to hilbert-space $\mathcal{B}$. In addition, $L(\mathcal{W})$ is used to denote $L(\mathcal{W}, \mathcal{W})$.

**Real world analysis** In this case, we need to describe the CPTP map $\Phi : L(\mathcal{W}) \to L(\mathcal{W} \otimes \mathcal{V} \otimes \mathcal{A} \otimes \mathcal{Y} \otimes \mathcal{Z})$ given by the $\mathbf{P} - \mathbf{V}$ protocol in the real interaction. If $H$ is the graph sent from $\mathbf{P}$ to $\mathbf{V}$ in its first message and then $\mathbf{V}$ measures bit $a$, then the state of registers $W, V, A$ changes from $\rho^{\mathcal{W}} |0\rangle\langle 0|^{\mathcal{VA}}$ to the normalized form of $(\mathbb{I}^{\mathcal{WV}} \otimes |a\rangle\langle a|^{\mathcal{A}}) V_H (\rho^{\mathcal{W}} \otimes |0\rangle\langle 0|^{\mathcal{VA}}) V_H^{\dagger} (\mathbb{I}^{\mathcal{WV}} \otimes |a\rangle\langle a|^{\mathcal{A}})$.

Consider the linear map $M_{H,a} \in L(\mathcal{W}, \mathcal{W} \otimes \mathcal{V})$ given by

$$M_{H,a} = (\mathbb{I}^{\mathcal{W}\mathcal{V}} \otimes \langle a|^{\mathcal{A}}) V_H (\mathbb{I}^{\mathcal{W}} \otimes |0\rangle^{\mathcal{V}\mathcal{A}})$$

The state of registers $W, V, A$ after **V** measures its bit can also be described as $M_{H,a}\rho^{\mathcal{W}} M_{H,a}^{\dagger} \otimes |a\rangle\langle a|^{\mathcal{A}}$. The complete CPTP map $\Phi : L(\mathcal{W}) \to L(\mathcal{W} \otimes \mathcal{V} \otimes \mathcal{A} \otimes \mathcal{Y} \otimes \mathcal{Z})$ is then given by (where we have substituted $H = \pi(G_0)$)

$$\Phi(X) = \frac{1}{n!} \sum_{\pi \in S_n} \sum_{a \in \{0,1\}} (M_{\pi(G_0),a} X M_{\pi(G_0),a}^{\dagger})^{\mathcal{W}\mathcal{V}} \otimes |a\rangle\langle a|^{\mathcal{A}} \otimes |\pi(G_0)\rangle\langle \pi(G_0)|^{\mathcal{Y}} \otimes |\pi\sigma^a\rangle\langle \pi\sigma^a|^{\mathcal{Z}}$$

**Simulated world** Our simulator would follow the same blueprint described earlier for the classical world simulator. We assume the simulator would use two additional registers - $R$ and $B$. $R$ is used to store the random choices the classical world simulator would have made. $B$ corresponds to the simulator's guess for the bit **V** is going to send.

Also, let $\mathcal{X} = \mathcal{V} \otimes \mathcal{A} \otimes \mathcal{Y} \otimes \mathcal{Z} \otimes \mathcal{R} \otimes \mathcal{B}$, i.e. the hilbert-space of all the registers except for register $W$. Also, let $|0\rangle^{\mathcal{X}}$ be the all-zero initial state of these registers. Before we formally define our simulator, we introduce some unitary operations which it would perform.

- Let $T \in L(\mathcal{Y} \otimes \mathcal{Z} \otimes \mathcal{R} \otimes \mathcal{B})$ be the unitary operator mapping the initial all-zero state of these registers to
$$\sum_{\pi,b} \frac{1}{\sqrt{2n!}} |\pi(G_b)\rangle^{\mathcal{Y}} \otimes |\pi\rangle^{\mathcal{Z}} \otimes |b\rangle^{\mathcal{B}} \otimes |\pi, b\rangle^{\mathcal{R}}$$
- Let $V \in L(\mathcal{Y} \otimes \mathcal{W} \otimes \mathcal{V} \otimes \mathcal{A})$ be the unitary operator which uses $Y$ as the control register to apply $V_H$ on registers $W, V, A$ when $Y$ is in the state $|H\rangle$. Therefore,
$$V = \sum_H V_H^{\mathcal{W}\mathcal{V}\mathcal{A}} \otimes |H\rangle\langle H|^{\mathcal{Y}}$$
- In addition to these, the simulator would be doing a projective measurement on registers $A, B$ (similar to the classical world simulator) to determine if they are the same bit. Let $\Pi_0 = |00\rangle\langle 00|^{\mathcal{A}\mathcal{B}} + |11\rangle\langle 11|^{\mathcal{A}\mathcal{B}}$ and $\Pi_1 = |01\rangle\langle 01|^{\mathcal{A}\mathcal{B}} + |10\rangle\langle 10|^{\mathcal{A}\mathcal{B}}$. $\{\Pi_0, \Pi_1\}$ is then a projective measurement which the simulator can perform. If the output of measurement is 0, then this would mean the simulator's guess for **V**'s bit was correct and it can easily complete the simulation. Otherwise, the simulator would need to do more work, as we show next.

Also, note that for notational convenience, the unitary operatos $T, V$ are used as if $V, T \in L(\mathcal{W} \otimes \mathcal{X})$ with identity operation on the remaining registers. With this we are now ready to formally describe the simulator. The same is given in Figure 1.2.

**Simulated world analysis** Lets now anlayze the CPTP map induced by the simulator's actions. We are going to do the following analysis assuming $W$ is in a pure state $|\psi\rangle$. The analysis for a general mixed state would follow similarly, and which we describe towards the end.

Let $|\gamma_0\rangle = |\psi\rangle^{\mathcal{W}}|0\rangle^{\mathcal{X}}$ denote the initial state of all the registers. The first thing the simulator does is apply the unitary transform $VT$. With this the state changes to

$$VT|\gamma_0\rangle = \frac{1}{\sqrt{2n!}} \sum_{\pi,b} |\pi(G_b)\rangle^{\mathcal{Y}} \otimes |\pi\rangle^{\mathcal{Z}} \otimes |b\rangle^{\mathcal{B}} \otimes |\pi, b\rangle^{\mathcal{R}} \otimes (V_H|\psi\rangle^{\mathcal{W}}|0\rangle^{\mathcal{V}\mathcal{A}})$$

Input: Auxiliary input register $W$.
Initial conditions: Registers $V, A, Y, Z, B, R$ are all initialized to the all-zero state.

- Perform unitary transform $T$, followed by $V$.
- Do a projective measurement described by $\{\Pi_0, \Pi_1\}$.
- If outcome of measurement is 0
  - Perform unitary transform $V^\dagger$, followed by $T^\dagger$.
  - Perform the unitary operation given by $\mathbb{I}^{\mathcal{W}} \otimes \sigma_z^{\mathcal{X}} = \mathbb{I}^{\mathcal{W}} \otimes (2|0\rangle\langle 0|^{\mathcal{X}} - \mathbb{I}^{\mathcal{X}})$.
  - Perform unitary transform given by $T$, followed by $V$.
- Halt and output registers $(W, V, A, Y, Z)$ (registers $B, R$ are traced out).

FIGURE 1.2: The simulator for GMW GI protocol

Following this the simulator applies the projective measurement given by $\{\Pi_0, \Pi_1\}$. Lets first look at the easier case, when the measurement outcome is 0. In this case, the CPTP map $\Psi : L(\mathcal{W}) \to L(\mathcal{W} \otimes \mathcal{V} \otimes \mathcal{A} \otimes \mathcal{Y} \otimes \mathcal{Z})$ is given by:

$$\begin{aligned}\Psi(|\psi\rangle\langle\psi|^{\mathcal{W}}) &= Tr_{B,R}\left(\Pi_0 VT|\gamma_0\rangle\langle\gamma_0|^{\mathcal{WX}} T^\dagger V^\dagger \Pi_0\right) \\ &= \frac{1}{2n!}\sum_{\pi,a}\Big((M_{H,a}|\psi\rangle\langle\psi|M_{H,a}^\dagger)^{\mathcal{WV}} \otimes |a\rangle\langle a|^{\mathcal{A}} \otimes |\pi(G_a\rangle\langle\pi(G_a|^{\mathcal{Y}} \otimes |\pi\rangle\langle\pi|^{\mathcal{Z}} \otimes |a\rangle\langle a|^{\mathcal{B}} \otimes |\pi, a\rangle\langle\pi, a|^{\mathcal{R}}\Big)\end{aligned}$$

Now note that $\pi \in S_n$ in the above summation can always be written as $\tau\sigma^a$, where $\sigma$ is the permutation defined in the original protocol s.t. $\sigma(G_1) = G_0$. Therefore, $\pi(G_a) = \tau\sigma_a(G_a) = \tau(G_0)$. Also, $H$ in the above summation is $\pi(G_a)$. Substituting, we get:

$$\Psi(|\psi\rangle\langle\psi|^{\mathcal{W}}) = \frac{1}{2n!}\sum_{\tau,a}\Big((M_{\tau(G_0),a}|\psi\rangle\langle\psi|M_{\tau(G_0),a}^\dagger)^{\mathcal{WV}} \otimes |a\rangle\langle a|^{\mathcal{A}} \otimes |\tau(G_0)\rangle\langle\tau(G_0)|^{\mathcal{Y}} \otimes |\tau\sigma^a\rangle\langle\tau\sigma^a|^{\mathcal{Z}}\Big)$$

Note that in this case the CPTP map induced is exactly the same as $\Phi$ in the real world. Lets look at the case now that the measurement outcome done by simulator was 1.

Recall from Figure 1.2 that in this case, the simulator applies the unitary transform $VT(\mathbb{I}^{\mathcal{W}} - \sigma_z^{\mathcal{X}})T^\dagger V^\dagger$ on all the registers.

For ease of analysis, lets define the projector $\Delta_0 = \mathbb{I}^{\mathcal{W}} \otimes |0\rangle\langle 0|^{\mathcal{X}}$ and $\Delta_1 = \mathbb{I}^{\mathcal{WX}} - \Delta_0$. The unitary $\mathbb{I}^{\mathcal{W}} \otimes \sigma_z^{\mathcal{X}}$ applied by the simulator is then equal to $\mathbb{I}^{\mathcal{W}} \otimes (2|0\rangle\langle 0|^{\mathcal{X}} - \mathbb{I}^{\mathcal{X}}) = \Delta_0 - \Delta_1$. We now have the following, which will help in our analysis. The proofs of these are ommitted from here and the reader is encouraged to check [3] for the detailed proof.

CLAIM 1.1. *The initial state $|\gamma_0\rangle$ is an eigen-vector of $\Delta_0 T^\dagger V^\dagger \Pi_0 VT\Delta_0$ with eigen-value of $\frac{1}{2}$.*

LEMMA 1.2. *Let $U, \Pi_0, \Pi_1, \Delta_0, \Delta_1 \in L(\mathcal{H})$ be linear operators on hilbert-space $\mathcal{H}$, where $U$ is a unitary and $\Pi_0, \Pi_1, \Delta_0, \Delta_1$ are projectors s.t. $\Pi_0 + \Pi_1 = \mathbb{I}$ and $\Delta_0 + \Delta_1 = \mathbb{I}$. Further suppose $|\gamma_0\rangle$ is an eigen-vector of $\Delta_0 U^\dagger \Pi_0 U \Delta_0$ with coresonding eigen-value $\lambda \in (0, 1)$. Define the following:*

$$|\delta_0\rangle = \frac{\Pi_0 U|\gamma_0\rangle}{\sqrt{\lambda}}, |\delta_1\rangle = \frac{\Pi_1 U|\gamma_0\rangle}{\sqrt{1-\lambda}}, |\gamma_1\rangle = \frac{\Delta_1 U^\dagger|\delta_0\rangle}{\sqrt{1-\lambda}}$$

*Then*

$$\begin{aligned}\langle\gamma_0|\gamma_1\rangle &= \langle\delta_0|\delta_1\rangle = 0\\ U|\gamma_0\rangle &= \sqrt{\lambda}|\delta_0\rangle + \sqrt{1-\lambda}|\delta_1\rangle\\ U|\gamma_1\rangle &= \sqrt{1-\lambda}|\delta_0\rangle - \sqrt{\lambda}|\delta_1\rangle\end{aligned}$$

Given the above Claim 1.1 and Lemma 1.2, we now show that the simulator produces the exact same CPTP map even in this case when the measurement outcome is 1. To do this, we use Lemma 1.2 with $|\gamma_0\rangle$ as the eigen-vector with eigen-value of $\frac{1}{2}$ and $U$ set as $VT$ (proven in Claim 1.1). We get the following from the equations:

$$\begin{aligned}|\delta_0\rangle = \sqrt{2}\Pi_0 VT|\gamma_0\rangle, |\delta_1\rangle &= \sqrt{2}\Pi_1 VT|\gamma_0\rangle, |\gamma_1\rangle = \sqrt{2}\Delta_1 (VT)^\dagger|\delta_0\rangle\\ VT|\gamma_0\rangle &= \frac{|\delta_0\rangle + |\delta_1\rangle}{\sqrt{2}}, VT|\gamma_1\rangle = \frac{|\delta_0\rangle - |\delta_1\rangle}{\sqrt{2}}\\ (VT)^\dagger|\delta_0\rangle &= \frac{|\gamma_0\rangle + |\gamma_1\rangle}{\sqrt{2}}, (VT)^\dagger|\delta_1\rangle = \frac{|\gamma_0\rangle - |\gamma_1\rangle}{\sqrt{2}}\end{aligned}$$

We are now ready to do our analysis. The initial state of all the registers is $|\gamma_0\rangle$, to which the simulator applies the unitary $VT$ and this changes the state to $\frac{|\delta_0\rangle+|\delta_1\rangle}{\sqrt{2}}$. Next, the projective measurement $\{\Pi_0, \Pi_1\}$ is applied. In case the outcome is 0, the state collapses to $|\delta_0\rangle$ and if the outcome is 1, then the state collapses to $|\delta_1\rangle$. We have already shown that in case the state collapses to state $|\delta_0\rangle$, the CPTP map output by the simulator is the same as the real world. In case the outcome is 1, to the collapsed state $|\delta_1\rangle$, the simulator applies the unitary $(VT)^\dagger$, which changes the state to $\frac{|\gamma_0\rangle-|\gamma_1\rangle}{\sqrt{2}}$. Thereafter, the simulator applies the unitary $\mathbb{I}^{\mathcal{W}} \otimes \sigma_z^{\mathcal{X}} = \Delta_0 - \Delta_1$. Now note that since $|\gamma_0\rangle$ was an eigen-vector of $\Delta_0 T^\dagger V^\dagger \Pi_0 VT\Delta_0$ and that $\Delta_0^2 = \Delta_0$ (since $\Delta_0$ is a projector), we have that $\Delta_0|\gamma_0\rangle = |\gamma_0\rangle$. Hence on applying $\Delta_0 - \Delta_1$, the state changes to $\frac{|\gamma_0\rangle+|\gamma_1\rangle}{\sqrt{2}}$. Finally, on applying the unitary $VT$, the state changes to $\frac{|\delta_0\rangle+|\delta_1\rangle+|\delta_0\rangle-|\delta_1\rangle}{2} = |\delta_0\rangle$, the same state to which the system had collapsed when the measurement outcome was 0 and which we have already shown produces the same CPTP map as the real-world.

Hence, with either the measurement of 0 or 1, we see that the CPTP map output by the simulator is the same as in the real world. Also, finally note that though the above analysis was done for a pure state $|\psi\rangle^{\mathcal{W}}$, since every mixed state can be written as a linear combination of pure states over $W$, the same analysis applies and the CPTP map output by the simulator is the same in all the cases.

## 1.4 Conclusion

In this report, we saw that the GMW graph isomorphism zero-knowledge protocol is also zero-knowledge against quantum polynomial-time adversaries. While we don't discuss this further, several other protocols which bear similarity to this can also be proven to be zero-knowledge using similar techniques. In particular, note that the simulator in the above construction gets rewinded itself as well when rewinding the verifier (when applying the unitary $T^\dagger V^\dagger$). This means this proof strategy is only useful for simulator constructions wherein the simulator doesn't need to use knowledge from different rewindings of the verifier. In particular, this proof strategy cannot be used for proofs of knowledge which commonly involve combining knowledge from different rewindings of the verifier. While there have been subsequent works which tackle this problem, we leave the taks of exploring them for future work.

## References

[1] O. Goldreich, S. Micali, and A. Wigderson. Proofs that yield nothing but their validity or all languages in np have zero-knowledge proof systems. *Journal of the ACM (JACM)*, 38(3):690–728, 1991.

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

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