# Quantum Automating $\mathbf{TC}^0$-Frege Is LWE-Hard

Noel Arteche
Lund University
noel.arteche@cs.lth.se

Gaia Carenini
École Normale Supérieure (ENS-PSL)
gaia.carenini@ens.psl.eu

Matthew Gray
University of Oxford
matthew.gray@cs.ox.ac.uk

**Abstract**

We prove the first hardness results against efficient proof search by quantum algorithms. We show that under Learning with Errors (LWE), the standard lattice-based cryptographic assumption, no quantum algorithm can weakly automate $\mathbf{TC}^0$-Frege. This extends the line of results of Krajíček and Pudlák (*Information and Computation*, 1998), Bonet, Pitassi, and Raz (*FOCS*, 1997), and Bonet *et al.* (*Computational Complexity*, 2004), who showed that Extended Frege, $\mathbf{TC}^0$-Frege and $\mathbf{AC}^0$-Frege, respectively, cannot be weakly automated by classical algorithms if either the RSA cryptosystem or the Diffie-Hellman key exchange protocol are secure. To the best of our knowledge, this is the first interaction between quantum computation and propositional proof search.

## 1 Introduction

Understanding the fundamental limits of efficient proof search is one of the main goals of proof complexity, a subfield of complexity theory. Classically, propositional proof complexity has been primarily concerned with proving lower bounds for the length of proofs in propositional proof systems, with the ultimate goal of settling whether **NP** = **coNP** [CR79]. In parallel, a growing line of research has focused on the computational hardness of finding propositional proofs.

Proof search is formally captured by the notion of *automatability*, introduced by Bonet, Pitassi, and Raz [BPR00]: a propositional proof system $S$ is automatable if there exists an algorithm that given as input a tautology $\varphi$, outputs an $S$-proof of $\varphi$ in time polynomial in the size of the shortest proof. By relating proofs and computation, automatability connects proof complexity to central areas of theoretical computer science such as automated theorem proving, SAT solving and combinatorial optimization [BN21], learning theory [PS22], or Kolmogorov complexity [Kra22].

Except for very weak proof systems like Truth Tables or Tree-Like Resolution (automatable in polynomial and quasipolynomial time [BP96; dRez21], respectively), most natural systems appear to be non-automatable under standard hardness assumptions. Existing hardness results can be split into two broad categories. Work from the late 90s and early 00s showed that stronger proof systems are non-automatable under cryptographic assumptions, while more recent work has showed that weaker proof systems are non-automatable under the optimal assumption that $\mathbf{P} \neq \mathbf{NP}$.

The cryptography-based approach was initiated by the seminal work of Bonet, Pitassi, and Raz [BPR00], which showed that $\mathbf{TC}^0$-Frege is hard to automate unless Blum integers can be factored by polynomial-size circuits. An earlier work of Krajíček and Pudlák [KP98] was then recast in the language of automatability, showing that Extended Frege is not automatable unless factoring can be solved efficiently. Finally, Bonet *et al.* [BDG+04] extended the result of [BPR00] from $\mathbf{TC}^0$-Frege to $\mathbf{AC}^0$-Frege under the stronger assumption that Blum integers cannot be factored by subexponential-size circuits.

Building on a long line of work [Iwa97; AB04; ABMP06; AR08; MPW19], the first **NP**-hardness result was shown in 2019, when Atserias and Müller [AM20] proved that Resolution is not automatable unless $\mathbf{P} = \mathbf{NP}$. This is optimal, as $\mathbf{P} = \mathbf{NP}$ implies the automatability of any proof system. Their proof uses a clever reduction from SAT that requires showing a specific lower bound for this system. The technique has since been adapted to other weak proof systems such as Regular and Ordered Resolution [Bel20], $k$-DNF Resolution [Gar20], Cutting

Planes [GKMP20], Nullstellensatz and Polynomial Calculus [dRGN+21], the OBDD proof system [IR22] and, more recently, even $\mathbf{AC}^0$-Frege [Pap23].

Though the latter works prove non-automatability under the optimal hardness assumption, their strength is incomparable to the cryptography-based results. The **NP**-hardness results all rely on proving specific super-polynomial proof complexity lower bounds for each system, meaning this strategy fails for $\mathbf{AC}^0[2]$-Frege and systems above, where no lower bounds are known. In contrast, the cryptographic hardness results work by ruling out *feasible interpolation* for these systems, a property which allows one to extract computational content from proofs. For a proof system $S$ proving its own soundness (such as $\mathbf{TC}^0$-Frege), feasible interpolation is equivalent to the notion of *weak* automatability introduced by Atserias and Bonet [AB04], the latter meaning that no proof system simulating $S$ is automatable. The question of whether weak systems such as Resolution are weakly automatable remains one of the major open problems in the field. In short, there exists a trade-off between the strength of the hardness assumption involved ($\mathbf{P} \neq \mathbf{NP}$ versus cryptographic) and the generality of the result (automatability versus weak automatability).

Our work is the first new contribution to the non-automatability of strong proof systems[1] in more than two decades. The early results in [KP98; BPR00; BDG+04] relied on the assumption that factoring is hard, which does not hold for quantum models of computation due to Shor's breakthrough algorithm [Sho94]. This raises the question of whether a quantum machine could carry out proof search efficiently for some strong proof system. Grover's search algorithm [Gro96] already provides a quadratic speed-up over brute-force proof search for any system. While this is not enough to achieve automatability, the possibility of more powerful algorithms motivates the interest in new conditional hardness results. The **NP**-hardness results outlined above imply that $\mathbf{NP} \not\subseteq \mathbf{BQP}$ suffices to rule out automatability for weak systems, but for stronger systems no widely believed assumption had yet been proven sufficient.

In this work, we formally define the notion of *quantum automatability* and show the first hardness results. We prove that $\mathbf{TC}^0$-Frege is not quantum automatable unless lattice-based cryptography can be broken by polynomial-size quantum circuits. Our results follow from the relationship between automatability and feasible interpolation suitably generalized to the quantum setting. This means that we also rule out quantum feasible interpolation and weak quantum automatability under the same cryptographic assumptions.

## Contributions

Our main contribution is proving the hardness of quantum automatability under the assumption that lattice-based cryptography is secure against quantum computers.

In 1996, Ajtai [Ajt96] gave the first worst-case to average-case reductions for lattice problems. In 1997, in joint work with Dwork [AD97], the worst-case hardness of such lattice problems was used to design public-key cryptosystems. Building on similar principles, the Learning with Errors (LWE) assumption of Regev [Reg09] has become the standard post-quantum cryptographic assumption. The LWE assumption is simple to state, surprisingly versatile, and does not seem susceptible to the core period-finding technique of Shor's algorithm.

In this work we show that any quantum algorithm that automates $\mathbf{TC}^0$-Frege can be used to break LWE.

**Theorem** (Main theorem, informal)**.** *If there exists a polynomial-time quantum algorithm that weakly automates $\mathbf{TC}^0$-Frege, then LWE can be broken in polynomial time by a quantum machine.*

We then exploit the simulation of $\mathbf{TC}^0$-Frege by $\mathbf{AC}^0$-Frege proofs of subexponential size to extend the result to $\mathbf{AC}^0$-Frege under a slightly stronger assumption, in the style of [BDG+04].

**Corollary.** *If there exists a polynomial-time quantum algorithm that weakly automates $\mathbf{AC}^0$-Frege, then LWE can be broken by subexponential-size quantum machines.*

In order to properly state and prove these results, we first formally define the notion of quantum automatability for quantum Turing machines. Note that a quantum algorithm might provide a wrong answer with small probability, so we need to be careful in choosing the right definitions. We show that our definition is equivalent to a similar one over uniform quantum circuits, and we verify that the definition is robust by reproving Impagliazzo's observation that weak automatability implies feasible interpolation, suitably translated to the quantum setting.

[1] We use the terms *weak* and *strong* informally throughout the paper. Traditionally, a strong proof system is a system that proves its own soundness, though it is often also intended to be a system for which lower bounds are lacking. For our purposes, *strong* refers to anything simulating $\mathbf{TC}^0$-Frege, for which both of the previous conditions apply.

## Techniques

The overall structure of the proofs follows the strategy of the previous non-automatability results of [KP98] and [BPR00], but the technical details are quite different due to certain complications arising from lattice-based cryptography. We outline below the main hurdles and the techniques used to overcome them.

**Quantum feasible interpolation.** Our result follows from conditionally ruling out feasible interpolation by quantum circuits. As observed by Impagliazzo, weak automatability implies feasible interpolation. We use this observation contrapositively. Suppose that a proof system can prove the injectivity of a candidate one-way function. In the presence of feasible interpolation, we are guaranteed the existence of small circuits capable of inverting the one-way function one bit at a time. If one believes in the security of the cryptographic object, one must conclude that the proof system does not admit feasible interpolation, and in turn that it is not weakly automatable.

For this strategy to work the candidate one-way function should fulfill two important conditions. First, its definition must be simple enough that the proof system can easily reason about it. For example, RSA requires modular exponentiation to be defined, which is conjectured not to be computable in $\mathbf{TC}^0$. This forced Bonet, Pitassi, and Raz to use instead the Diffie-Hellman protocol in [BPR00]. Second, the candidate one-way function must be perfectly injective. The rather technical reason for perfect injectivity is that feasible interpolation allows one to carry out the inversion bit by bit, which does not guarantee retrieving a correct preimage if there are multiple ones.

A few perfectly injective one-way functions based on lattice geometry have been proposed throughout the literature, e.g., see [PW08; GKVW20; MP12]. However, we consider instead a simple scheme for worst-case lattice-based functions that closely resembles the one described in [Mic11]. Such a scheme has the advantage that its injectivity can be easily verified, and that its worst case one way-ness is guaranteed by the assumed hardness of Learning with Errors, which we will now discuss.

**Learning with Errors and certificates of injectivity.** We base our construction directly on the Learning with Errors assumption. The assumption is simple to define: roughly speaking, it conjectures that a vector $x$ multiplied by some public matrix $A$, together with some Gaussian noise, $Ax + \varepsilon$, is hard to recover. While the most naive functions based on LWE are not necessarily injective, we can bound the magnitude of the error vectors to construct a family of functions where almost all of the functions are injective. For most matrices $A$, the corresponding function $f_A$ in this family is worst-case one-way assuming the hardness of LWE [Mic11].

However, the functions being injective and worst-case one-way is not sufficient, because their injectivity needs to provable inside $\mathbf{TC}^0$-Frege. Unlike with the Diffie-Hellman construction in [BPR00], where a single proof showed the injectivity of the protocol for all generators, here each injective $f_A$ may require a tailored proof of injectivity. Fortunately, most of these $f_A$ can have their injectivity certified by a left-inverse of $A$ together with a short basis for the dual lattice of the $q$-ary lattice spanned by $A$. These short bases not only certify injectivity, they can be used as a trapdoors to invert the function [Pei+16]. Though we do not exploit this directly, one may think of the automating algorithm as extracting such trapdoors from proofs. Essentially, these certificates can be exploited to prove the injectivity of most $f_A$ inside $\mathbf{TC}^0$-Frege.

With these properties we can show that feasible interpolation can be used to invert almost all $f_A$, which is sufficient to break LWE and its associated worst-case lattice problems.

**Formal theories for linear algebra.** The most technical component of the previous work on $\mathbf{TC}^0$-Frege and $\mathbf{AC}^0$-Frege in [BPR00; BDG+04] consisted in formalizing many basic properties of arithmetic directly inside the propositional proof systems, which can be quite cumbersome. While we can borrow a large part of the existing formalization in [BPR00], putting it together to carry out arguments about lattice geometry is still quite convoluted.

Instead, we follow the approach of Krajíček and Pudlák, who showed the injectivity of RSA in Extended Frege by reasoning in Buss's theory $S^1_2$ of bounded arithmetic. Universal theorems of this first-order theory translate into propositional tautologies with succinct proofs in Extended Frege. For $\mathbf{TC}^0$-Frege and its sequent calculus formalism PTK, the corresponding first-order theory of bounded arithmetic is the two-sorted theory $\mathrm{VTC}^0$ introduced by Cook and Nguyen [CN10]. This theory is quite expressive and can reason even about analytic

functions, as shown by Jeřábek [Jeř23]. However, since we are mostly interested in statements of matrix algebra, we use the more convenient formal theory LA for linear algebra introduced by Soltys and Cook [SC04].

The theory LA is quantifier-free and operates directly with matrices. It is strong enough to prove their ring properties, but weak enough to allow all theorems in LA to translate into propositional tautologies with short $\mathbf{TC}^0$-Frege proofs. In order to handle all the concepts required in our arguments, we work over a conservative extension of LA over the rationals which we show still propositionally translates into $\mathbf{TC}^0$-Frege.

## Open problems

To the best of our knowledge, this is the first interaction between quantum computation and propositional proof search, and we believe connections between the two fields are worthwhile exploring further. We outline below three open lines of research, ranging from the interaction between quantum computation and proof complexity to a classical problem in the theory of automatability.

**Positive results?** While hardness of proof search in most natural proof systems is now conditionally ruled out under different assumptions, there exists a handful of systems for which no non-automatability results are known. This is the case of the Res($\oplus$), Res(log), Sherali-Adams or Sum-of-Squares proof systems. Could quantum algorithms automate any of these systems efficiently?

Even for proof systems where worst-case hardness is known, could quantum algorithms provide a significant speed-up over brute-force search? Clearly, Grover's algorithm achieves already a quadratic speed-up, but could this be pushed further in some cases?

**Quantum proof complexity.** Hardness results in automatability involve three key elements: the proof system, the hardness assumption and the model of computation for the automating algorithm. In this work we shifted the latter two to the quantum setting, by choosing a post-quantum cryptographic assumption and a quantum model of computation, but the proof systems considered remained classical.

What would it mean to have an inherently quantum proof system? In the same way that Extended Frege can be seen as **P**/poly-Frege, could we define a proof system where lines are quantum circuits? This could open the door to a quantum analogue of the Cook-Reckhow program, where showing lower bounds on quantum proof systems would be related to the question of whether **QMA** = **coQMA**. We note that an analogous approach exists in the field of parameterized complexity, starting with the work of Dantchev, Martin, and Szeider [DMS11], who defined parameterized proof complexity as a program to gain evidence on the **W**-hierarchy being different from **FPT**. As an intermediate step, it would make sense to consider the case of randomized proof systems and the relationship between **MA** and **coMA**, though this has proven to be challenging so far.

We remark that while Pudlák [Pud09] already defined the notion of quantum derivation rules for propositional proof systems and defined the quantum Frege proof system, his approach is orthogonal to ours, in that those systems are still designed to derive propositional tautologies. In fact, he showed that classical Frege systems simulate quantum Frege systems, though classical Frege proofs cannot be extracted from quantum proofs by a classical algorithm unless factoring is in **FP**.

**Towards generic hardness assumptions.** Our results, like the original works in [KP98; BPR00; BDG+04], require *concrete* cryptographic assumptions. That is, we assume that some specific candidate one-way function or cryptographic protocol is secure. The reason is that in order to obtain the upper bounds on which to apply feasible interpolation we need concrete formulas to manipulate inside the different proof systems.

A major open problem in the theory of automatability is to disentangle these results from concrete families of candidate one-way functions. That is, can we prove that $\mathbf{TC}^0$-Frege is not (weakly) automatable under the assumption that, say, one-way functions exist? Even better, can one obtain **NP**-hardness of automating strong proof systems without the need to prove lower bounds first, in a way different from the strategy of Atserias and Müller [AM20]? This seems to require conceptual breakthroughs.

## Structure of the paper

The paper is structured as follows. Section 2 recalls the necessary concepts in proof complexity and lattice-based cryptography needed in the rest of the paper. Section 3 defines automatability for quantum Turing machines

and uniform quantum circuits and proves the equivalence between both models to then reprove Impagliazzo's observation on the relation between automatability and feasible interpolation, now in the quantum setting. Section 4 states and proves the main theorem of the paper. The section first presents a detailed overview of the main argument, while the subsections contain all the necessary technical work.

# 2 Preliminaries

We assume basic familiarity with computational complexity theory, propositional logic and quantum circuits. We review the main concepts needed from proof complexity and refer the reader to standard texts like [Kra19] for further details. We also recall some relevant notions from linear algebra and lattice geometry useful in our arguments.

## 2.1 Proof complexity

Following Cook and Reckhow [CR79], a *propositional proof system* $S$ for the language TAUT of propositional tautologies is a polynomial-time surjective function $S : \{0,1\}^* \to \text{TAUT}$. We think of $S$ as a proof checker that takes some proof $\pi \in \{0,1\}^*$ and outputs $S(\pi) = \varphi$, the theorem that $\pi$ proves. Soundness follows from the fact that the range is exactly TAUT, and implicational completeness is guaranteed by the fact that $S$ is surjective. One may alternatively define proof systems for refuting propositional contradictions. We move from one setup to the other depending on context.

We denote by $\text{size}_S(\varphi)$ the *size* of the smallest $S$-proof of $\varphi$ plus the size of $\varphi$. We say that a proof system $S$ is *polynomially bounded* if for every $\varphi \in \text{TAUT}$, $\text{size}_S(\varphi) \leq |\varphi|^{O(1)}$. We say that a proof system $S$ *polynomially simulates* a system $Q$ if for every $\varphi \in \text{TAUT}$, $\text{size}_S(\varphi) \leq \text{size}_Q(\varphi)^{O(1)}$. For a family $\{\varphi_n\}_{n\in\mathbb{N}}$ of propositional tautologies, we write $S \vdash \varphi_n$ whenever $\text{size}_S(\varphi_n) \leq |\varphi|^{O(1)}$. Finally, a proof system $S$ is said to be *closed under restrictions* if whenever $S$ proves a formula $\varphi$ in size $s$, for every partial restriction $\rho$ to the variables in $\varphi$, there exists a proof of the restricted formula $\varphi_{\restriction\rho}$ in size $s^{O(1)}$.

The focus of this work is on a specific class of proof systems known as *Frege systems*. A Frege system is a finite set of axiom schemas and inference rules that are sound and implicationally complete for the language of propositional tautologies built from the Boolean connectives negation ($\neg$), conjunction ($\wedge$), and disjunction ($\vee$). A Frege proof is a sequence of formulas where each formula is obtained by either substitution of an axiom schema or by application of an inference rule on previously derived formulas. As long as the set of inference rules is finite, sound and implicationally complete, the specific choice of rules does not effect the size of the proofs up to polynomial factors, as all Frege systems polynomially simulate each other [Kra19, Theorem 4.4.13].

We can make gradations between Frege systems by restricting the complexity of their proof lines. For a circuit class $C$, the system $C$-Frege is any Frege system where lines are restricted to be $C$-circuits (see [Jer05] for a formal definition). In this setup, a standard Frege system amounts to $\mathbf{NC}^1$-Frege. We are mostly interested in the weaker systems $\mathbf{AC}^0$-Frege and $\mathbf{TC}^0$-Frege, where the proof lines are, respectively, circuits of constant-depth and unbounded fan-in, and threshold circuits of constant-depth and unbounded fan-in. A *threshold circuit* is a Boolean circuit where gates can be the usual $\neg, \vee, \wedge$ as well as the threshold ones $\text{Th}_k(x_1, \ldots, x_n)$, where $\text{Th}_k$ is true if at least $k$ of its input are true.

It is often convenient to consider an alternative formalism of $\mathbf{TC}^0$-Frege in the style of Gentzen's sequent calculus. The *Propositional Threshold Calculus* PTK [CN10, Chapter X.4.1] is a version of the propositional sequent calculus where the cuts are restricted to threshold formulas of constant depth. We refer to [BPR00] for a complete rendering of the derivational rules of PTK.

## 2.2 Lattice geometry

We recall some basic definitions from lattice geometry. For a linearly independent set of $n$ vectors $\mathcal{B} = \{b_1, \ldots, b_n\} \subseteq \mathbb{R}^m$, which we often treat simply as an $m \times n$ matrix, the *lattice* over $\mathcal{B}$ is defined to be the set of all integer linear combinations of vectors in $\mathcal{B}$,

$$\mathcal{L}(\mathcal{B}) := \{x \in \mathbb{R}^m \mid \text{there is } a \in \mathbb{Z}^n \text{ such that } x = \mathcal{B}a\}.$$

When the vectors in $\mathcal{B}$ belong in $\mathbb{Z}_q^m$ for some modulus $q$, we can further define a *modular lattice* over $\mathcal{B}$, denoted $\mathcal{L}_q(\mathcal{B})$, to be the set of all integer linear combinations of the basis modulo $q$,

$$\mathcal{L}_q(\mathcal{B}) \coloneqq \{x \in \mathbb{Z}_q^m \mid \text{there is } a \in \mathbb{Z}_q^n \text{ such that } \mathcal{B}a \equiv x \bmod q\},$$

where the mod function is applied element-wise in the vector.

We define the length of a vector $x$ in $\mathcal{L}_q(\mathcal{B})$ to be the Euclidean norm of the shortest vector in $\mathbb{Z}^m$ that is congruent to $x$ modulo $q$. Note that these shortest vectors will always fall in the domain $[-\lfloor q/2 \rfloor, \lfloor q/2 \rfloor]^m$.

A *$q$-ary lattice* $\Delta_q(\mathcal{B})$ can be thought of as an extension of a modular lattice back to $\mathbb{Z}^m$ and is the set of all vectors $x \in \mathbb{Z}^m$ congruent to members of the modular lattice,

$$\Delta_q(\mathcal{B}) \coloneqq \{x \in \mathbb{Z}^m \mid \text{there is } a \in \mathbb{Z}^n \text{ such that } \mathcal{B}a \equiv x \bmod q\}.$$

Note that because for all $x \in \{0, q\}^m$, $x \in \Delta_q(\mathcal{B})$, we have that all $q$-ary lattices have rank $m$.

From the definitions above it is clear that $\mathcal{L}_q(\mathcal{B}) \subseteq \Delta_q(\mathcal{B})$. Consequently a proof that no vector in $\Delta_q(\mathcal{B})$ has length less than $\ell$ also proves that no vector in $\mathcal{L}(\mathcal{B})$ has length less than $\ell$.

Another important concept in lattice geometry is that of a *dual lattice*. Given a lattice $\mathcal{L}(\mathcal{B})$, its dual lattice $\mathcal{L}^*(\mathcal{B})$ is defined to be the set of vectors within the subspace spanned by $\mathcal{B}$ whose inner product with any element in $\mathcal{L}$ is an integer. Formally,

$$\mathcal{L}^*(\mathcal{B}) \coloneqq \{y \in \mathbb{R}^m \mid \text{there is } z \in \mathbb{R}^n \text{ such that } y = \mathcal{B}z \text{ and for all } x \in \mathcal{L}(\mathcal{B}), \langle x, y \rangle \in \mathbb{Z}\},$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product. The dual lattice is also a lattice, whose base admits a closed form.

**Lemma 2.1.** *For a base $\mathcal{B} \in \mathbb{R}^{m \times n}$, $\mathcal{L}^*(\mathcal{B}) = \mathcal{L}(\mathcal{B}(\mathcal{B}^\intercal \mathcal{B})^{-1})$.*

Note that, if $\mathcal{B} \in \mathbb{Z}^{m \times n}$, it is easy to show that $\mathcal{B}(\mathcal{B}^\intercal \mathcal{B})^{-1} \in \mathbb{Q}^{m \times n}$, and, therefore, that any $x \in \mathcal{L}^*(\mathcal{B})$ belongs to $\mathbb{Q}^m$. This lemma is standard and can be found, for example, in [Mic11].

Modular lattices and $q$-ary lattices are fairly different mathematical objects, but we can show that given a matrix $\mathcal{B} \in \mathbb{Z}_q^{m \times n}$ such that $\operatorname{rank}(\mathcal{B}) = n$, there exists a closed form for a matrix $\mathcal{B}'$ such that $\mathcal{L}(\mathcal{B}') = \Delta_q(\mathcal{B})$.

**Lemma 2.2** (Full-rank modular lattices have $q$-ary lattice bases)**.** *Let $\mathcal{B} \in \mathbb{Z}_q^{m \times n}$ and define $C \in \{0,1\}^{m \times m}$ to be the permutation matrix that swaps the appropriate rows so that the first $n$ rows of $C\mathcal{B}$ are linearly independent. Let $\mathcal{B}_1 \in \mathbb{Z}^{n \times n}$ and $\mathcal{B}_2 \in \mathbb{Z}^{m-n \times n}$ be matrices such that $\mathcal{B} = [C\mathcal{B}_1 \mid C\mathcal{B}_2]^\intercal$. Then, for $\mathcal{B} \in \mathbb{Z}_q^{m \times n}$, if $\operatorname{rank}(\mathcal{B}) = n$, $\Delta_q(\mathcal{B}) = \mathcal{L}(\mathcal{B}')$, where*

$$\mathcal{B}' = C \begin{bmatrix} I_n & 0 \\ (C\mathcal{B})_2 (C\mathcal{B})_1^{-1} & qI_{m-n} \end{bmatrix} C^{-1},$$

*and where the inverses $M^{-1}$ are defined over the modular lattice $\mathbb{Z}_q^m$.*

Note that we can combine this corollary with Lemma 2.1 to get a closed form for $\mathcal{B}'$ such that $\Delta_q^*(\mathcal{B}) = \mathcal{L}(\mathcal{B}')$.

The *$i$-th successive minimum* of a lattice $\mathcal{L}$ is $\lambda_i(\mathcal{L}) \coloneqq \inf\{r \in \mathbb{Z} \mid \dim(\operatorname{span}(\mathcal{L} \cap B(0, r))) \geq i\}$, where $B(0, r)$ is the ball of radius $r$ around the origin. Roughly speaking, this means that $\lambda_i(\mathcal{L})$ is the length of the $i$-th smallest linearly independent vector in the lattice.

There exists an intimate relationship between a lattice and its dual, as captured by Banaszczyk's Transference Theorem.

**Theorem 2.3** (Transference Theorem [Ban93])**.** *For any rank-$n$ lattice $\mathcal{L} \subseteq \mathbb{Z}^m$, $1 \leq \lambda_1(\mathcal{L}) \cdot \lambda_n(\mathcal{L}^*) \leq n$.*

Modular lattices $\mathcal{L}_q(\mathcal{B})$ are subsets of $\mathbb{Z}_q^m$, not $\mathbb{Z}^m$, and therefore the Transference Theorem does not directly apply. However we are able to leverage the fact that $\lambda_1(\Delta_q(\mathcal{B})) = \min(q, \lambda_1(\mathcal{L}_q(\mathcal{B})))$ to indirectly apply it through the $q$-ary lattice.

We recall useful properties of random lattices.

**Lemma 2.4.** *For a randomly selected matrix $A \in \mathbb{Z}_q^{m \times n}$, we have that*

*(i)* $\Pr_A[\operatorname{rank}(A) = n] \geq n/q^{m-n+1}$;

*(ii)* $\Pr_A[\lambda_1(\mathcal{L}_q(A)) < r \mid \operatorname{rank}(A)] \leq (2r+1)^m / q^{m-2n-1}$.

These properties are folklore. For the sake of completeness, we provide proofs in Appendix C.

## 2.3 Learning with Errors (LWE)

Learning with Errors (LWE) is a central problem of learning theory, introduced by Regev [Reg09].

**Assumption 2.5** (The Learning with Errors (LWE) assumption [Reg09; Pei+16])**.** Let $m = n^{O(1)}$, $q \leq 2^{n^{O(1)}}$, let $s \sim \mathbb{Z}_q^n$ be a secret vector, $A \sim \mathbb{Z}_q^{m \times n}$, and $\varepsilon \in \mathbb{Z}_q^m$ a sample from the discrete Gaussian with standard deviation $\alpha q$ with $\alpha = o(1)$ and $\alpha \in [0, 1]$. The *Learning with Errors assumption* states that there is no quantum inverter $M$ running in time $n^{O(1)}$ such that $M(A, As+\varepsilon)$ outputs with noticeable probability some $s'$ such that $As' + \varepsilon = As + \epsilon$, where the probability is over the choice of $s$, $A$, $\varepsilon$, and the internal randomness of $M$.

The security of this assumption relies on the existence of worst-case to average-case reductions to fundamental lattice problems conjectured to be hard. In particular, as shown by Regev [Reg09], breaking LWE implies solving the $\gamma$-GapSVP problem for an approximation factor $\gamma = n^2$. Here, $\gamma$-GapSVP refers to the *$\gamma$-Approximate Shortest Vector Problem*: given a lattice basis $\mathcal{B} \in \mathbb{Q}^{m \times n}$ and a distance threshold $r > 0$, decide whether $\lambda_1(\mathcal{L}(\mathcal{B})) \leq r$, or $\lambda_1(\mathcal{L}(\mathcal{B})) > \gamma r$, when one of those cases is promised to hold.

The belief that $\gamma$-GapSVP is intractable is backed by the fact that the problem is **NP**-hard under randomized reductions when the approximation factor is constant [Ajt96; Pei+16; BP23]. However, for the range of $\gamma$ in which the reduction to LWE works, no **NP**-hardness is known. Obtaining **NP**-hardness for polynomial approximation factors would imply the breakthrough consequence of basing cryptography on worst-case hardness assumptions. In turn, this would turn our non-automatability results into **NP**-hardness results. As appealing as this might be, it is unlikely. For $\gamma \geq \sqrt{n}$, the problem $\gamma$-GapSVP is known to be in $\mathbf{NP} \cap \mathbf{coNP}$ [AR05] and thus cannot be **NP**-hard unless **PH** collapses.

## 2.4 The formal theory LA

The theory LA is a quantifier-free theory introduced by Soltys and Cook [SC04] whose main objects are matrices. This is not technically speaking a first-order theory of bounded arithmetic like those used in [KP98], but like them it admits a propositional translation into Frege systems.

The system LA operates over three sorts: *indices* (intended to be natural numbers), *field elements* (over some abstract field $\mathbb{F}$), and *matrices* (with entries over $\mathbb{F}$). Variables for these three sorts are usually denoted $i, j, k, \ldots$ for indices, $a, b, c, \ldots$ for field elements, and $A, B, C, \ldots$, for matrices. We sometimes use lower-case letters $v, w, \ldots$ for vectors, which are seen as a special case of matrices.

The language of LA consists of the following constant, predicate and function symbols, over the three different sorts:

- Index sort: $0_{\text{index}}$, $1_{\text{index}}$, $+_{\text{index}}$, $\cdot_{\text{index}}$, $-_{\text{index}}$, $\mathsf{div}$, $\mathsf{rem}$, $\mathsf{cond}_{\text{index}}$, $\leq_{\text{index}}$, $=_{\text{index}}$
- Field sort: $0_{\text{field}}$, $1_{\text{field}}$, $+_{\text{field}}$, $\cdot_{\text{index}}$, $-_{\text{index}}$, $^{-1}$, $\mathsf{r}$, $\mathsf{c}$, $\mathsf{e}$, $\Sigma$, $\mathsf{cond}_{\text{field}}$, $=_{\text{field}}$
- Matrix sort: $=_{\text{matrix}}$

The meaning of the symbols is the standard one, except for $-_{\text{index}}$ that denotes the cutoff subtraction ($i - j = 0$ if $i < j$) and for $a^{-1}$, denoting the inverse of a field element $a$, with $0^{-1} = 0$. For operations over matrices, $\mathsf{r}(A)$ and $\mathsf{c}(A)$ are, respectively, the number of rows and columns in $A$, $\mathsf{e}(A, i, j)$ is the field element $A_{i,j}$ (with $\mathsf{e}(A, i, j) = 0$ if either $i = 0$, $j = 0$, $i > \mathsf{r}(A)$ or $j > \mathsf{c}(A)$) and $\Sigma A$ is the sum of the elements in $A$. The function symbol $\mathsf{cond}(\alpha, t_1, t_2)$ is interpreted to mean that if $\alpha$ holds, then the returned value should be $t_1$, else $t_2$, where $\alpha$ is a formula all of whose atomic subformulas have the form $m \leq n$ or $m = n$, where $m$ and $n$ are of the index sort, and $t_1, t_2$ are terms either both of index sort or both of field sort.

The language of LA can be enriched with the following defined terms: index maximum ($\mathsf{max}$), matrix sum ($+$, when sizes of the matrices are compatible), scalar product ($\cdot$), matrix transpose ($A^\intercal$), zero ($0$) and identity matrices ($I$), matrix trace ($\mathrm{tr}$), dot product ($\langle \_, \_ \rangle$), and matrix product ($\cdot$). See [SC04, Section 2.1] for details on the definitions of these terms. In general, whenever it is clear from context, we drop the subscripts indicating the sort and we use standard linear algebra notation for the sake of readability.

The theory then consists of several groups of axioms fixing the meaning of these symbols. These are rather lengthy to state, so we relegate them to Appendix A.1, where we also include several theorems derived by Soltys and Cook inside LA.

Observe that the theory is field-independent, but whenever we fix the field to be either finite or $\mathbb{Q}$, LA has the robust property that every theorem translates into a family of propositional formulas with short $\mathbf{TC}^0$-Frege

proofs. This is the main property of LA that we shall exploit, which corresponds to Theorem 6.3 in their original paper.

# 3 Quantum automatability and feasible interpolation

Following [BPR00], we say that a propositional proof system $S$ is *automatable in time $t$* if there exists a deterministic Turing machine $A$ that on input a formula $\varphi$ outputs an $S$-proof of $\varphi$, if one exists, in time $t(\text{size}_S(\varphi))$. We now consider the possibility of replacing $A$ by a probabilistic or quantum Turing machine. The main issue in the definition is now that the output of the machine may be erroneous, albeit with small probability. Note, however, that if a machine were to output an incorrect proof, we would be able to easily detect this, since we can verify the proofs in polynomial time. We may thus assume that when yielding an incorrect proof, the machine will restart and find another one. Hence, instead of asking for the error-probability of the machine to be bounded, we ask for the expected running time to be bounded. The following definition captures this idea.

**Definition 3.1** (Quantum and randomized automatability)**.** Let $S$ be a propositional proof system and let $t : \mathbb{N} \to \mathbb{N}$ be a time-constructible function. We say that $S$ is *quantum* (respectively, *random*) *automatable in time $t$* or simply *quantumatable in time $t$* if there exists a quantum Turing machine (respectively, a randomized Turing machine) that on input a formula $\varphi$ outputs an $S$-proof of $\varphi$, if one exists, in expected time $t(\text{size}_S(\varphi))$.

In what follows, we assume $t$ to be a polynomial and talk simply about a system being *automatable* or *quantum automatable*, without reference to $t$. Since quantum circuits are often more convenient than quantum Turing machines, we also define automatability in the circuit setting.

**Definition 3.2** (Circuit automatability)**.** Let $S$ be a propositional proof system. We say that $S$ is *circuit-automatable* if there exists a constant $c$ and a uniform multi-output circuit family $\{C_{n,s}\}_{n,s\in\mathbb{N}}$ of size $(n+s)^c$ such that $C_{n,s}$ takes as input a formula $\varphi$ of size $n$ and outputs an $S$-proof of size $s^c$ if a proof of size $s$ exists, and is allowed to output any string otherwise.

The generalization to randomized and quantum circuits is now immediate.

**Definition 3.3.** Let $S$ be a propositional proof system. We say $S$ is *quantum circuit-automatable* if there exists a constant $c$ and a uniform multi-output quantum circuit family $\{C_{n,s}\}_{n,s\in\mathbb{N}}$ of size $(n+s)^c$ such that $C_{n,s}$ takes as input a formula $\varphi$ of size $n$, and outputs an $S$-proof of size $s^c$ with probability at least $2/3$ if a proof of size $s$ exists, and is allowed to output any string otherwise. We say that $S$ is *random circuit-automatable* if the circuit is classical but also takes as input a sequence $r$ of random bits and, for at least $2/3$ of the choices for $r$, $C_{n,s}(\varphi, r)$ outputs an $S$-proof of size $s^c$ if a proof of size $s$ exists, and is allowed to output any string otherwise.

In fact, the machine-based and circuit-based definitions are equivalent.

**Proposition 3.4.** *Let $S$ be a propositional proof system. The following equivalences hold:*

*(i) the system $S$ is automatable if and only if it is circuit-automatable;*

*(ii) the system $S$ is random automatable if and only if it is random circuit-automatable;*

*(iii) the system $S$ is quantum automatable if and only if it is quantum circuit-automatable.*

We defer the rather simple proof to Appendix B.

Even if a proof system is not automatable, one might still hope for an algorithm that finds some proof efficiently, even if it is in a different proof system. We say that a proof system $S$ is *weakly automatable* if there exists another proof system $Q$ and an algorithm $A$ that given a formula $\varphi$, outputs a $Q$-proof of $\varphi$ in time $\text{size}_S(\varphi)^{O(1)}$. The concept was introduced by Atserias and Bonet [AB04], who further showed that this is equivalent to $S$ being simulated by a system $Q$ that is itself automatable. Despite the fact that weak automatability has been studied from several perspectives, e.g. [AM11; HP11; BPT14], establishing whether weak proof systems—such as Resolution—are weakly automatable remains one of the main open problems in the area. It is straightforward to extend the notion of weak automatability to the quantum setting.

Weak automatability is closely related to feasible interpolation. We recall this connection in its classical form and then move to the quantum setting.

**Definition 3.5** (Feasible interpolation [Kra97; Pud97])**.** We say that a proof system $S$ has the *feasible interpolation property* if there exists a polynomial-time computable function $I$ such that for every tautological split formula $\varphi(x, y, z) = \alpha(x, z) \vee \beta(z, y)$, whenever a proof $\pi$ in $S$ derives $\varphi$ in size $s$, $I(\pi)$ produces an *interpolant circuit* $C_\varphi$ of size $s^{O(1)}$ that takes as input an assignment $\rho$ to the $z$-variables and such that

$$C_\varphi(\rho) = \begin{cases} 0 & \text{only if } \alpha(x, \rho) \text{ is a tautology} \\ 1 & \text{only if } \beta(\rho, y) \text{ is a tautology} \end{cases}$$

indicating which side of the conjunction is tautological.

In [BPR00] the following observation relating (weak) automatability and feasible interpolation was attributed to Impagliazzo. We refer to it as *Impagliazzo's observation.*

**Proposition 3.6** (Impagliazzo's observation [BPR00, Thm. 1.1])**.** *If a proof system is weakly automatable and closed under restrictions, then it admits feasible interpolation.*

Impagliazzo's observation is useful contrapositively: to rule out (weak) automatability it suffices to rule out feasible interpolation, as done in [KP98] and [BPR00]. We outline this strategy further in Section 4, where we instantiate it together with our cryptographic assumption.

To use feasible inerpolation in our setting, we suitably adapt the definition to the quantum world.

**Definition 3.7** (Quantum feasible interpolation)**.** We say that a proof system $S$ has the *quantum feasible interpolation property* if there exists a polynomial-time computable function $I$ such that, for every tautological split formula $\varphi(x, y, z) = \alpha(x, z) \vee \beta(z, y)$, whenever a proof $\pi$ derives $\varphi$ in $S$ in size $s$, $I(\pi)$ prints the description of a quantum interpolant circuit $C_\varphi$ of size $s^{O(1)}$ as in Definition 3.5 with probability at least $2/3$. If the circuit is instead randomized, we call this property *random feasible interpolation.*

Interestingly, feasible interpolation is not affected by moving from classical automatability to randomized automatability. This is essentially folklore, but we reprove it for the sake of completeness.

**Proposition 3.8.** *If a proof system $S$ is weakly random automatable and closed under restrictions, then it has feasible interpolation by deterministic Boolean circuits.*

*Proof.* The proof is essentially the same as the original proof in [BPR00], except for having to take randomness into account. Suppose $R$ is a probabilistic automating algorithm for $S$. By Proposition 3.4.(ii), we can instead think of a family of randomized circuits $\{C_{n,s}\}_{n,s\in\mathbb{N}}$ that, for some fixed constant $c$, outputs proofs of size $s^c$ when a proof of size $s$ exists. Furthermore, let $d$ be the constant in the exponent that bounds the blow-up in size happening in the closure under restrictions. Given a split formula $\varphi = \alpha \vee \beta$, we want to obtain an interpolant circuit $C_\varphi$.

Use the automating algorithm to find some proof of $\varphi$. Let $s_0$ be the size of such a proof. We first show that it is possible to extract a polynomial-size randomized circuit that computes the interpolant with one-sided error. Consider the circuit that takes as input the restriction $\rho$ together with some random bits and proceeds to compute $C_{|\alpha|,s_0^d}(\alpha_{\restriction\rho}, r)$. If this circuit finds a proof of $\alpha_{\restriction\rho}$ and it is checked to be correct, we output 0; else, we output 1. We claim that for at least $2/3$ choices of $r$, this circuit is a correct interpolant (and, in fact, whenever it outputs 0, it is always correct). First, note that if we output 0 it is because a proof of $\alpha_{\restriction\rho}$ was found, in which case it is correct to say that $\alpha_{\restriction\rho}$ is a tautology. Otherwise, we will always output 1. The only problematic case is when the circuit outputs 1 while $\neg\beta_{\restriction\rho}$ is satisfiable. If such was the case, then let $\sigma$ be a satisfying assignment to the $z$-variables such that $\neg\beta_{\restriction\rho,\sigma}$ is satisfied. Since $S$ can prove $\varphi$ in size $s_0$ and $S$ is closed under restrictions, we know that $S$ can prove $\varphi_{\restriction\rho,\sigma}$ in size $s_0^d$, and this proof must clearly be deriving $\alpha_{\restriction\rho,\sigma} = \alpha_{\restriction\rho}$. Since $s_0^{cd} \geq s_0^d$, for a "good" choice of $r$ the circuit $C_{|\alpha|,s_0^d}(\alpha_{\restriction\rho}, r)$ would have found such a proof, so the only reason why we could have output 1 is that we chose a bad $r$. But this of course only happens with probability at most $1/3$. So this randomized circuit interpolates $\varphi$, makes only one-sided error, and has size polynomial in the shortest proof.

We now replicate the strategy used in Adleman's theorem ($\mathbf{BPP} \subseteq \mathbf{P}/\mathrm{poly}$) to show that in fact randomness is not needed in the circuit. One can follow here the standard argument as presented, for example, in [AB09, Thm. 7.15]: given the interpolant circuit $F_\varphi$, perform error reduction and then argue that there must be a string of random bits that is "good" for all inputs of the same size. The circuit no longer makes mistakes and computes $f_\varphi$ as desired. □

*Remark* 3.9 (Constructive feasible interpolation). Our definition of feasible interpolation deviates from the one given in standard texts like that of Krajíček [Kra19], and follows instead the one given by Pudlák [Pud03], who imposes the condition that the interpolant circuit must be constructed from the given proof in polynomial time. Note that even if we adopted the non-constructive definition, the kind of feasible interpolation obtained by the construction above achieves this property anyway.

The constructivity requirement is useful to obtain a sort of converse of Impagliazzo's observation: if a propositional proof system has uniform polynomial-size proofs of its reflection principle, then it is weakly automatable (see [Pud03, Prop. 3.6]).

Since randomness does not buy us anything when it comes to proof search, all hardness results immediately transfer to the randomized setting. In particular, for every proof system $S$ simulating $\mathbf{TC}^0$-Frege, $S$ is not weakly random automatable unless Blum integers can be factored by polynomial-size randomized circuits. For weak proof systems where automatability is known to be **NP**-hard, the systems cannot be automatable unless $\mathbf{NP} \subseteq \mathbf{BPP}$.

When moving to the quantum setting, unfortunately, we do not know of any way to get a deterministic circuit for the interpolant. Instead, we have the following natural version of Impagliazzo's observation.

**Proposition 3.10.** *If a proof system is quantum automatable and closed under restrictions, then it admits feasible interpolation by quantum circuits.*

*Proof.* The proof follows the argument in Proposition 3.8, except we can no longer apply the final step to get rid of quantumness. The interpolant now is a quantum circuit, since it is simulating the quantum circuit $C_{|\alpha|,s_0^d}(\alpha_{\restriction\rho})$. □

# 4 $\mathbf{TC}^0$-Frege is hard to quantum automate

The quantum version of Impagliazzo's observation (Proposition 3.10) is the main tool needed for our hardness results, which we are now ready to state formally.

**Theorem 4.1** (Main theorem)**.** *Let $S$ be a proof system simulating $\mathbf{TC}^0$-Frege. If $S$ is weakly quantum automatable, then the LWE assumption fails.*

We then extend the result to $\mathbf{AC}^0$-Frege under a stronger assumption. This is done by applying the fact that $\mathbf{TC}^0$-Frege proofs can be translated into $\mathbf{AC}^0$-Frege proofs of subexponential size (see, for example, Theorems 2.5.6 and 18.7.3 in [Kra19] or the original work on the non-automatability of $\mathbf{AC}^0$-Frege [BDG+04]).

**Corollary 4.2.** *Let $S$ be a proof system simulating $\mathbf{AC}^0$-Frege. If $S$ is weakly quantum automatable, then the LWE assumption is broken by a quantum algorithm running in time $2^{n^{o(1)}}$.*

We devote the rest of the paper to formally proving Theorem 4.1.

Suppose $h : \{0,1\}^n \to \{0,1\}^n$ is an injective and secure one-way function. Let $x$, $y$ and $z$ denote variables ranging over $\{0,1\}^n$ and assume that $\mathbf{TC}^0$-Frege is able to state and refute efficiently the following unsatisfiable formula,

$$(h(x) = z \wedge x_1 = 0) \wedge (h(y) = z \wedge y_1 = 1)\,,$$

where $x_1, y_1$ are respectively the first bit of $x$ and $y$. The unsatisfiability follows precisely from the fact that $h$ is injective, and hence every output has a unique preimage.

If $\mathbf{TC}^0$-Frege admits feasible interpolation, we are guaranteed the existence of a small circuit $C(z)$ such that

$$C(z) = \begin{cases} 0 & \text{if } h(x) = z \wedge x_1 = 0 \text{ is unsatisfiable} \\ 1 & \text{if } h(y) = z \wedge y_1 = 1 \text{ is unsatisfiable} \end{cases}$$

meaning that $C$ is able to invert one bit of $z$. Since every output has a unique preimage, we can iterate the process to get the entire input string. This contradicts the assumption that $h$ is one-way.

In order to instantiate the proof strategy to rule out quantum feasible interpolation, we now need a candidate one-way function that is injective and conjectured to be post-quantum secure and for which injectivity can be proven inside the proof system. Unfortunately, to the best of our knowledge, no such candidate function

is currently known, or not with enough security guarantees[2]. Alternatively, we may use other cryptographic objects that do achieve some form of injectivity, such as bit commitments, but the formalization of the latter does not seem simpler than the approach we follow instead. We now explain how we avoid this issue.

The most reliable post-quantum cryptographic assumptions have their security based on worst-case reductions to lattice problems conjectured to be hard. This is the case of the Learning with Errors framework [Reg09], on which we base the security of the following class of candidate one-way functions. For these functions, as well as the basic properties of them that we employ, we follow the treatment of Micciancio [Mic11]. We include the details for the proof complexity readers, who may not be familiar with these constructions.

**Definition 4.3** (The candidate functions $f_A$)**.** Let $m = n^{O(1)}$, $q \leq 2^{n^{O(1)}}$, and $c = \alpha q/\sqrt{n}$, where $\alpha \in [0,1]$. For every matrix $A \in \mathbb{Z}_q^{m\times n}$, we define the function $f_A : \mathbb{Z}_q^n \times \{\varepsilon \in \mathbb{Z}_q^m : |\varepsilon| \leq 10c\sqrt{mn}\} \to \mathbb{Z}_q^m$ as

$$f_A(s,\varepsilon) \coloneqq (As + \varepsilon) \bmod q\,.$$

At this point, we would like to show inside $\mathbf{TC}^0$-Frege that the conjunction

$$(f_A(x) = z \wedge x_1 = 0) \wedge (f_A(y) = z \wedge y_1 = 1) \tag{1}$$

is a contradiction, where $A$ is represented by free variables and $x_1$ and $y_1$ refer to the first bits of $x$ and $y$. Unfortunately, the problem concerning injectivity mentioned above remains. The formula is not necessarily a contradiction, since for some choices of $A$, $f_A$ is not injective. We can show, however, that with high probability over the choice of $A$, the function $f_A$ will satisfy two conditions that imply injectivity. Namely, $A$ will be full rank and the shortest vector in the $q$-ary lattice spanned by $A$ will be large enough.

The following proposition, which captures this idea, is standard. We reprove it here for the sake of completeness, since we shall formalize part of it inside the proof systems later.

**Proposition 4.4.** *Let $n \in \mathbb{N}, m = n\log n$ and $q \geq n^5$. With high probability over the choice of $A \in \mathbb{Z}_q^{m\times n}$, $\operatorname{rank}(A) = n$ and $\lambda_1(\mathcal{L}_q(A)) > 20c\sqrt{nm}$. Furthermore, when these hold, the function $f_A$ is injective.*

*Proof.* From Lemma 2.4.i we get that $\Pr_{A\sim\mathcal{U}(\mathbb{Z}_q^{m\times n})}[\operatorname{rank}(A) < n] \leq n/q^{m-n+1}$ . By Lemma 2.4.ii we can see that

$$\Pr_{A\sim\mathcal{U}(\mathbb{Z}_q^{m\times n})}[\lambda_1(\mathcal{L}(A)) \leq 20c\sqrt{mn} \mid \operatorname{rank}(A) = n] \leq \frac{(40c\sqrt{mn}+1)^m}{q^{m-2n-1}}\,.$$

The probability that a random $A$ does not satify the conditions in the statement is at most the sum of the two probabilities above, which are both negligible for our choice of $m$ and $q$.

For injectivity, suppose for contradiction that there exist $x, x', \varepsilon, \varepsilon'$, with either $x \neq x'$ or $\varepsilon \neq \varepsilon'$, causing a collision $f_A(x,\varepsilon) = Ax + \varepsilon = Ax' + \varepsilon' = f_A(x',\varepsilon')$. We have two cases.

(a) If $\varepsilon = \varepsilon'$, then the collision happens if and only if $\operatorname{rank}(A) < n$, which contradicts the assumption.

(b) Suppose that $\varepsilon \neq \varepsilon'$. We have that $\varepsilon - \varepsilon' = A(x' - x)$. Since the norm of $\varepsilon - \varepsilon'$ is at most $20c\sqrt{nm}$, by transitivity we have that the length of $A(x' - x)$ is bounded by the same quantity. However, the latter belongs to the lattice and therefore we obtain a contradiction. □

Luckily for us, these two conditions are succinctly certifiable! Indeed, to certify that the matrix $A$ is full rank we may provide a left-inverse $A_L^{-1}$ such that $A_L^{-1}A = I_n$. Unfortunately, we cannot guarantee that all injective $f_A$ have simple certificates of the second property, $\lambda_1(\mathcal{L}_q(A)) > 20c\sqrt{nm}$. Nevertheless, we show in Section 4.2 that almost all of them do. These certificates take the form of sets $W = \{w_1, \ldots, w_m\} \subseteq \Delta_q^*(A)$ of short linearly independent vectors in the dual of the $q$-ary lattice. We prove—using the left inequality of Banaszczyk's Transference Theorem—that such a set suffices to certify the second property, and then show—using the right side of Banaszczyk's Transference Theorem— that the certificate $W$ exists with high probability.

[2] In a previous version of this work we formalized the injectivity of several group-based post-quantum cryptographic assumptions, such as MOBS [RS21], as well as variants of supersingular isogeny-based Diffie-Hellman protocols, which unfortunately all happen to be now broken more or less efficiently.

**Definition 4.5** (Certificate of injectivity)**.** A *certificate of injectivity* for the function $f_A$, with $A \in \mathbb{Z}_q^{m\times n}$, is a pair $(A_L^{-1}, W)$ such that $A_L^{-1}$ is a left-inverse so that $A_L^{-1}A = I_n$, and $W = \{w_1, \ldots, w_m\} \subseteq \Delta_q^*(A)$ is a set of $m$ linearly independent vectors such that $\max_{i\in[m]} ||w_i|| < 1/20c\sqrt{nm}$.

**Proposition 4.6.** *Let $n \in \mathbb{N}$, $m = n \log n$, $q = n^{10}$, and $A \in \mathbb{Z}_q^{m\times n}$. The following hold:*

*(i) if there is a certificate of injectivity $(A_L^{-1}, W)$ for $f_A$, then $f_A$ is injective;*

*(ii) if $\mathrm{rank}(A) = n$ and $\lambda_1(\mathcal{L}_q(A)) > 20mc\sqrt{nm}$, then there exists a certificate of injectivity for $f_A$;*

*(iii) with high probability over the choice of $A$, $\mathrm{rank}(A) = n$ and $\lambda_1(\mathcal{L}_q(A)) > 20mc\sqrt{nm}$.*

Observe that given a certificate $(A_L^{-1}, W)$, verifying its correctness is a rather simple task: it is sufficient check that $A_L^{-1}A = I_n$, to verify that $W$ is a set of linearly independent vectors in $\Delta_q^*(A)$, and finally to ensure that the vectors in $W$ are small enough.

Let us return to the propositional system. We denote by $\textsc{Inj}(f_A)$ the propositional formula encoding that $f_A$ is injective. From this formula, $\mathbf{TC}^0$-Frege can derive that (1) is a contradiction. However, $\textsc{Inj}(f_A)$ is false if we leave $A$ as free variables. We instead prove $\textsc{Inj}(f_{A_0})$ for concrete injective $f_{A_0}$, where $A_0$ is hardwired. The concrete $f_{A_0}$ for which we do it are the ones that admit a certificate of injectivity.

Essentially, we formalize inside $\mathbf{TC}^0$-Frege that a certificate of injectivity implies injectivity. That is,

$$\mathbf{TC}^0\text{-Frege} \vdash \textsc{Cert}(C_A) \to \textsc{Inj}(f_A)\,, \tag{2}$$

where $\textsc{Cert}(C_A)$ encodes that $C_A$ is a correct certificate for $f_A$. Here $C_A$ and $A$ are free variables. This implication is precisely Proposition 4.6.i above. The proof inside the system is carried out in Section 4.3.

Now, given a concrete certificate $C_{A_0}$ for $f_{A_0}$, the formula $\textsc{Cert}(C_{A_0})$ is derivable inside $\mathbf{TC}^0$-Frege, which amounts to the system verifying the certificate's correctness. From this, $\mathbf{TC}^0$-Frege proves $\textsc{Inj}(f_{A_0})$.

The rest of this section completes the missing parts in the proof. Section 4.1 sketches the known fact that $f_A$ is worst-case one-way based on the hardness of Learning with Errors, while Section 4.2 proves Proposition 4.6 showing the existence of certificates. We remark that the arguments and techniques are standard in cryptography and readers familiar with the area might want to skip them. We include them for the sake of completeness and to cater for the proof complexity reader that may have never come across these ideas before, and we refer to standard texts like [Mic11] for further details. Finally, Section 4.3 formalizes the certificate-to-injectivity implication above inside the theory $\mathsf{LA}_{\mathbb{Q}}$, which propositionally translates into $\mathbf{TC}^0$-Frege. Section 4.4 reconstructs the final argument.

## 4.1 Security of $f_A$

The functions in $\{f_A\}_{A\in\mathbb{Z}_q^{m\times n}}$ very closely resemble the standard Learning with Errors functions, the only difference being that we have set a maximum value on the magnitude of the error vectors and allowed these to be chosen as a uniform part of the input (instead of being sampled from a Gaussian distribution). We now observe that inverting these functions allows us to invert LWE with high probability over the choice of the error vector.

**Lemma 4.7** ([Mic11])**.** *Suppose there exists an algorithm $B$ taking as input $A \in \mathbb{Z}_q^{m\times n}$ and a string $z$ and outputting a preimage in $f_A^{-1}(z)$ with probability $p$. Then, LWE can be broken with probability $0.99p$ over the choice of the error vector $\varepsilon$ and the internal randomness of $B$.*

*Proof.* It suffices to show that with high probability, the error vectors in the standard Learning with Errors functions are bounded as in our definition of $f_A$, and thus the same inverter for $f_A$ will also work for most of the original LWE instances. This follows from a standard Gaussian tail bound. Thus, if we are able to invert $f_A$ on all outputs with probability $p$, then we are able to invert its corresponding LWE function with probability, say, $0.99p$ over the choice of $\varepsilon$. □

Note that it is in fact possible to invert with all but negligible probability, since finding a vector whose norm is far above the expectation with high probability requires that several independently sampled coordinates all return values much larger than the expected one. For simplicity, we chose to prove a weaker result which suffices for our applications.

## 4.2 Existence of certificates of injectivity: Proof of Proposition 4.6

This section proves the three statements of Proposition 4.6.

**Proposition 4.6.i** (Correctness of certificates)**.** *If there is a certificate of injectivity* $(A_L^{-1}, W)$ *for* $f_A$*, then* $\operatorname{rank}(A) = n$ *and* $\lambda_1(\mathcal{L}_q(A)) > 20c\sqrt{nm}$*.*

*Proof.* As discussed in the proof of Proposition 4.4, $f_A$ is injective if and only if both $\operatorname{rank}(A) = n$ and $\lambda_1(\mathcal{L}_q(A)) > 20c\sqrt{mn}$. By elementary linear algebra, $\operatorname{rank}(A) = n$ if and only if there exists a $A_L^{-1}$. As previously observed we know that $\lambda_1(\Delta_q(A)) = \min(q, \lambda_1(\mathcal{L}(A)))$ and since $20c\sqrt{mn} \leq q/m$, therefore it suffices to show that the existence of $W$ as described above implies that $\lambda_1(\Delta_q(A)) > 20c\sqrt{mn}$.

Because it is a $q$-ary lattice we known that $\operatorname{rank}(\Delta_q(A)) = m$. By rearranging the left inequality of the Transference Theorem for rank-$m$ lattices, we get that $\lambda_1(\Delta_q(A)) \geq 1/\lambda_m(\Delta_q^*(A))$. By the definition of $W$, we conclude that $\lambda_m(\mathcal{L}^*) < 1/20c\sqrt{nm}$, which implies that $\lambda_1(\mathcal{L}_q(A)) = \lambda_1(\Delta_q(A)) > 20c\sqrt{nm}$. □

**Proposition 4.6.ii** (Conditional existence of certificates)**.** *If* $\operatorname{rank}(A) = n$ *and* $\lambda_1(\mathcal{L}_q(A)) > 20mc\sqrt{nm}$*, then there exists a certificate of injectivity for* $f_A$*.*

*Proof.* Since we assumed that $\operatorname{rank}(A) = n$, there exists a left inverse $A_L^{-1}$ for $A$. It therefore suffices to show that if $\operatorname{rank}(A) = n$ and $\lambda_1(\mathcal{L}_q(A)) > 20mc\sqrt{nm}$, then there exists a set of vectors $W$ satisfying the conditions above.

By the right inequality of the Transference Theorem for rank-$m$ lattices, we get that $\lambda_m(\Delta_q^*(A)) \leq m/\lambda_1(\Delta_q(A))$. Since $\lambda_1(\mathcal{L}_q(A)) > 20mc\sqrt{nm} \leq q$, and $\lambda_1(\Delta_q(A)) = \min(q, \lambda_1(\mathcal{L}_q(A))) = \lambda_1(\mathcal{L}_q(A))$ there must exist a set of $m$ linearly independent vectors in $\Delta_q^*(A)$, such that $\max_{i\in[m]} ||w_i|| < 1/20c\sqrt{nm}$. □

**Proposition 4.6.iii** (Existence of certificates with high probability)**.** *Let* $n \in \mathbb{N}$, $m = n \log n$, $q \geq n^5$, $c \leq \sqrt{nm}/40$ *and* $A \in \mathbb{Z}_q^{m\times n}$ *be sampled uniformly at random. The probability that* $\operatorname{rank}(A) = n$ *and* $\lambda_1(\mathcal{L}_q(A)) > 20cm\sqrt{nm}$ *is at least*

$$1 - \frac{n}{q^{m-n+1}} - m^{3m-(m-2n-1)\log_m q}.$$

*This probability is greater than exponentially close to* 1 *for our choice of* $q$ *and* $m$*.*

*Proof.* For the following equations we define $\mathrm{E}_{\text{short}}$ to be the event that $\lambda_1(\mathcal{L}_q(A)) \leq 20cm\sqrt{nm}$.

$$\begin{aligned}\Pr_A[\operatorname{rank}(A) \neq n \vee \mathrm{E}_{\text{short}}] &= \Pr_A[\operatorname{rank}(A) \neq n] + \Pr_A[\mathrm{E}_{\text{short}} \wedge \operatorname{rank}(A) = n] \\ &\leq \Pr_A[\operatorname{rank}(A) \neq n] + \Pr_A[\mathrm{E}_{\text{short}} \mid \operatorname{rank}(A) = n]\,.\end{aligned}$$

By Lemma 2.4.i, we know that $\Pr_A[\operatorname{rank}(A) \neq n] \leq n/q^{m-n+1}$, and by the second point of Lemma 2.4.ii, we have that

$$\Pr[\mathrm{E}_{\text{short}} \mid \operatorname{rank}(A) = n] \leq \frac{(40mc\sqrt{mn})^m}{q^{m-2n-1}} \leq \frac{(m^2n)^m}{q^{m-2n-1}} \leq \frac{m^{3m}}{m^{(m-2n-1)\log_m q}} = m^{3m-(m-2n-1)\log_m q}\,.$$

□

## 4.3 Formalization

At this point, the only thing left is the formalization of the implication $\textsc{Cert}(C_A) \to \textsc{Inj}(f_A)$ inside the propositional system. Since this is rather cumbersome, we work instead in the more convenient theory LA of linear algebra of Soltys and Cook [SC04]. The theory, however, is field-independent, which means we cannot state or prove properties about the ordering of the rationals, which is needed in our arguments. Furthermore, we sometimes use the fact that certain matrices are over the integers, so we must be able to identify certain elements as integers. To solve this, we introduce a conservative extension of the theory, called $\mathsf{LA}_{\mathbb{Q}}$, which assumes the underlying field to be $\mathbb{Q}$.

### 4.3.1 The conservative extension $\mathsf{LA}_{\mathbb{Q}}$

On top of the exiting symbols of the language of LA, we have two new predicate symbols int and $<_{\mathbb{Q}}$. The symbol $<_{\mathbb{Q}}$, which we overload onto $<$ in what follows, is intended to represent the usual ordering relation over the rationals. As usual, we write $x \leq y$ to mean $x < y \vee x = y$. Recall that equality of field elements was a symbol in the base theory LA. The int predicate, applied to a field element $q$, written $\mathsf{int}(q)$, is supposed to be true whenever the rational $q$ is an integer.

We extend the axiom-set of LA with axioms for the new symbols, as follows. Recall that the original axioms of LA are listed in Appendix A.1.

**Axioms for** int

$(\mathrm{Int}_1)$ $\mathsf{int}(x) \wedge \mathsf{int}(y) \rightarrow \mathsf{int}(x + y)$

$(\mathrm{Int}_2)$ $\mathsf{int}(x) \wedge \mathsf{int}(y) \rightarrow \mathsf{int}(x \cdot y)$

$(\mathrm{Int}_3)$ $\mathsf{int}(x) \wedge 0 < x \rightarrow 1 \leq x$

**Axioms for** $<_{\mathbb{Q}}$

$(\mathrm{Ord}_1)$ $x \leq x \wedge \neg(x < x)$

$(\mathrm{Ord}_2)$ $x \leq y \wedge y \leq x \rightarrow x = y$

$(\mathrm{Ord}_3)$ $x \leq y \wedge y \leq z \rightarrow x \leq z$

$(\mathrm{Ord}_4)$ $x \leq y, z \leq w \rightarrow x + z \leq y + w$

It might seem that these axioms are not enough to fix a reasonable interpretation of the symbols. Indeed, the axioms for int only really force that addition and multiplication are closed under this predicate and they do not identify $\mathbb{Z}$ as a substructure of $\mathbb{Q}$. The reason this is not an issue is that we are only interested in $\mathsf{LA}_{\mathbb{Q}}$ for its propositional translation. For our purposes, these axioms are the only ones we need to prove the required claims about lattices, and once we translate to the propositional setting, the symbols will take the standard intended interpretation, so it does not matter that these are underspecified in the theory.

It is not hard to show that theorems of $\mathsf{LA}_{\mathbb{Q}}$ admit succinct $\mathbf{TC}^0$-Frege proofs. Showing this requires extending the propositional translation in [SC04] to include the new symbols and axioms. We do this in Appendix A.2.

### 4.3.2 Formalization of the proofs

We are ready to present the formal proofs needed inside our theory. In what follows, LA_._ stands for the corresponding axiom in Appendix A.1.

We start by formalizing inside $\mathsf{LA}_{\mathbb{Q}}$ the classical Cauchy-Schwartz inequality, which is later needed in one of the proofs.

**Lemma 4.8** (Cauchy-Schwartz in $\mathsf{LA}_{\mathbb{Q}}$)**.** *The theory* $\mathsf{LA}_{\mathbb{Q}}$ *proves that for every* $u, v \in \mathbb{Q}^n$, $\langle u, v\rangle^2 \leq \langle u, u\rangle \cdot \langle v, v\rangle$.

*Proof.* We show that $\mathsf{LA}_{\mathbb{Q}}$ can derive the following equality:

$$\frac{1}{\langle v, v\rangle}\langle(\langle v, v\rangle u - \langle u, v\rangle v), (\langle v, v\rangle u - \langle u, v\rangle v)\rangle = \langle u, u\rangle\langle v, v\rangle - \langle u, v\rangle^2 . \tag{3}$$

We do this explicitly by deriving the following chain of equalities:

$$\begin{aligned}
&\frac{1}{\langle v, v\rangle}\langle(\langle v, v\rangle u - \langle u, v\rangle v), (\langle v, v\rangle u - \langle u, v\rangle v)\rangle = && \text{(LA7.b)}\\
&\frac{1}{\langle v, v\rangle}(\langle(\langle v, v\rangle u - \langle u, v\rangle v), (\langle v, v\rangle u)\rangle + \langle(\langle v, v\rangle u - \langle u, v\rangle v), (-\langle u, v\rangle v)\rangle) = && \text{(LA7.b + LA7.c + LA7.a)}\\
&\frac{1}{\langle v, v\rangle}(\langle v, v\rangle^2\langle u, u\rangle - \langle v, v\rangle\langle v, u\rangle^2) = && \text{(LA7.c)}\\
&(\langle v, v\rangle\langle u, u\rangle - (\langle v, u\rangle^2)
\end{aligned}$$

Observe that the left-hand side of Equation (3) is positive and $\mathsf{LA}_{\mathbb{Q}}$ can derive this fact since squares are positive in this theory. Therefore, the right-hand side of Equation (3) is positive as well. From this the inequality follows. □

The other technical component needed in the final proof is a weakening of the lower bound in Banaszczyk's Transference Theorem (see Theorem 2.3). Informally, we need to prove that for every $A \in \mathbb{Q}^{m\times n}$, every non-zero vector $v \in \mathcal{L}(A)$ and any set of linearly independent vectors $W = \{w_1, \ldots, w_n\} \subseteq \mathcal{L}^*(A)$, $\langle v, v\rangle \cdot \langle w_i, w_i\rangle \geq 1$.

In order for $\mathsf{LA}_{\mathbb{Q}}$ to process the conditions of the theorem, we provide certificate-like objects ensuring all the different hypotheses. For example, when we quantify over a vector $v$ belonging to a lattice $\mathcal{L}(A)$, we provide the vector of coefficients $c_v$ such that $Ac_v = v$. Note as well that when we quantify over matrices with elements in $\mathbb{Z}$, we are using the int predicate under the hood to enforce the entries to be integers.

**Lemma 4.9** (Banaszczyk's left inequality in $\mathsf{LA}_{\mathbb{Q}}$)**.** *The theory $\mathsf{LA}_{\mathbb{Q}}$ proves the following implication. Let $A \in \mathbb{Z}^{m\times n}$, $B \in \mathbb{Q}^{n\times n}$, $v \in \mathbb{Q}^n$, $c_v \in \mathbb{Z}^m$, $W = [w_1|\ldots|w_n] \in \mathbb{Q}^{m\times n}$, $c_W = [c_{w_1} \mid \cdots \mid c_{w_n}] \in \mathbb{Z}^{m\times n}$, $W' \in \mathbb{Q}^{m\times n}$ fulfilling the following conditions:*

1. *the vector $v$ is non-zero, $v \neq 0_n$;*
2. *the vector $v$ belongs to the lattice $\mathcal{L}(A)$, $v = Ac_v$;*
3. *the vectors in $W$ belong to the dual lattice[3] $\mathcal{L}^*(A)$, $w_i = ABc_{w_i}$ for all $i \in [n]$;*
4. *$(A^\intercal A)B = I_n$;*
5. *the vectors in $W$ are linearly independent, $W'W^\intercal = I_n$.*

*Then, for some $i \in [n]$, $\langle v, v\rangle \cdot \langle w_i, w_i\rangle \geq 1$.*

*Proof.* The proof has two steps. First, we show that for all $i \in [n]$, $\langle v, w_i\rangle \in \mathbb{Z}$. To do this we use the following chain of equalities, where $w$ is some arbitrary column $w_i$ of $W$, and where the comments on the side refer to either axioms of $\mathsf{LA}_{\mathbb{Q}}$ or the assumptions in the statement of the lemma:

$$\begin{aligned}
\langle v, w\rangle &= \langle Ac_v, ABc_w\rangle && \text{(by ass. 2 and 3)}\\
&= c_v^\intercal A^\intercal ABc_w && \text{(by def. of dot product)}\\
&= c_v^\intercal (A^\intercal A)Bc_w && \text{(by associativity, LA5.i)}\\
&= c_v^\intercal c_w. && \text{(by ass. 4)}
\end{aligned}$$

By assumption, the factors of $c_v^\intercal c_w$ have integer entries, and by the closure under integer multiplication ($\mathrm{Int}_2$), we deduce that for all $i \in [n]$, $\langle v, w_i\rangle \in \mathbb{Z}$.

In the second step of the proof, we show that there is $j \in [n]$ such that $\langle v, w_i\rangle \neq 0$. We consider the vector $s \coloneqq W^\intercal v$. Note that by definition, the $i$-th entry of $s$ is $\langle w_i, v\rangle$. We can multiply both sides by the same matrix $W'$, leading to $W's = W'W^\intercal v$. Using associativity (LA5.i), assumption (5) and properties of the identity matrix (LA5.f), we get that $W's = v$. Suppose that for all $j \in [n]$, $\langle v, w_j\rangle = 0$. Then, by definition, $s = 0_n$. We can easily derive (using LA3.a, LA3.c and LA3.i) that $W's = 0$ and therefore $v = 0$. This contradicts assumption (5).

We now recall that by axiom ($\mathrm{Int}_3$) of $\mathsf{LA}_{\mathbb{Q}}$, every non-zero positive integer is greater or equal than 1. This implies that for some $i \in [n]$, $\langle v, w_j\rangle \geq 1$. Using Cauchy-Schwartz (Lemma 4.8) for this $i$, we get $1 \leq \langle v, w_i\rangle^2 \leq \langle v, v\rangle \cdot \langle w_i, w_i\rangle$. □

We are now ready to prove in $\mathsf{LA}_{\mathbb{Q}}$ that a correct certificate of injectivity implies the injectivity of $f_A$. Informally, we aim to prove that given a certificate of injectivity as in Definition 4.5, the function $f_A$ is injective. As before, we need to provide some additional objects together with the certificate to make sure $\mathsf{LA}_{\mathbb{Q}}$ can reason about this conditional implication and carry out the verification of the certificate.

**Lemma 4.10** (Certificate-implies-injectivity in $\mathsf{LA}_{\mathbb{Q}}$)**.** *Let $A \in \mathbb{Z}^{m\times n}$, $B \in \mathbb{Q}^{n\times n}$, $v_1 \in \mathbb{Q}^n$, $c_{v_1} \in \mathbb{Z}^m$, $v_2 \in \mathbb{Q}^n$, $c_{v_2} \in \mathbb{Z}^m$, $\varepsilon_1 \in \mathbb{Q}^n$, $\varepsilon_2 \in \mathbb{Q}^n$, $W = [w_1|\ldots|w_n] \in \mathbb{Q}^{m\times n}$, $c_W = [c_{w_1} \mid \cdots \mid c_{w_n}] \in \mathbb{Z}^{m\times n}$, $W' \in \mathbb{Q}^{m\times n}$ fulfilling the following conditions:*

1. *the vector $v_1$ belongs to the lattice $\mathcal{L}(A)$, $v_1 = Ac_{v_1}$;*
2. *the vector $v_2$ belongs to the lattice $\mathcal{L}(A)$, $v_2 = Ac_{v_2}$;*
3. *the vectors $v_1$ and $v_2$ are distinct, $v_1 \neq v_2$;*

[3] This dual lattice, in fact, admits a closed form for its base, as in Lemma 2.1. In particular, $B$ can be seen as $(A^\intercal A)^{-1}$.

4. *the vectors in* $W$ *belong to the dual lattice* $\mathcal{L}^*(A)$, $w_i = ABc_{w_i}$ *for all* $i \in [n]$;
5. $(A^\intercal A)B = I_n$;
6. *the vectors in* $W$ *are linearly independent,* $W'W^\intercal = I_n$;
7. $\langle w_i, w_i \rangle < 1/400c^2nm$;
8. $\langle \varepsilon_1 - \varepsilon_2, \varepsilon_1 - \varepsilon_2 \rangle < 400c^2nm$.

*Then,* $Av_1 + \varepsilon_1 \neq Av_2 + \varepsilon_2$.

*Proof.* The proof proceeds by contradiction. Suppose that $Av_1 + \varepsilon_1 = Av_2 + \varepsilon_2$, meaning that a collision exists in the range of $f_A$. By simple algebraic manipulations in $\mathsf{LA}_{\mathbb{Q}}$, we derive that $A(v_1 - v_2) = \varepsilon_2 - \varepsilon_1$. The left-hand side belongs to the lattice and thus satisfies that $\langle A(v_1 - v_2), A(v_1 - v_2) \rangle \geq \lambda_1(\mathcal{L}(A))^2$. Applying Lemma 4.9 to $v := A(v_1 - v_2)$, we deduce that there exists $i$ such that $\langle v, v \rangle \geq 1/\langle w_i, w_i \rangle$. From assumption (7), we have that $\langle v, v \rangle > 400c^2nm$. We can now easily obtain a contradiction with assumption (8). □

## 4.4 Proof of Theorem 4.1

We prove the theorem for $\mathbf{TC}^0$-Frege; it then easily follows that the same holds for any stronger system. Suppose that $\mathbf{TC}^0$-Frege is weakly quantum automatable, that is, suppose that $S$ is a quantum automatable proof system simulating $\mathbf{TC}^0$-Frege. Let $Q$ be the quantum algorithm automating $S$. We describe a quantum algorithm $Q'$ that takes as input a matrix $A$ defining a function $f_A$ as in Definition 4.3 and an output $z$ of this function and succeeds in finding a preimage of $z$ with high probability.

For a specific input matrix $A_0$, consider the formula $\textsc{Cert}(C_A) \to \textsc{Inj}(f_A)$, where $C$ and $A$ are free variables. In Lemma 4.10 this implication was proven inside $\mathsf{LA}_{\mathbb{Q}}$, and by the propositional translation for $\mathsf{LA}_{\mathbb{Q}}$ in Theorem A.2 we get and efficient proof inside $\mathbf{TC}^0$-Frege, and thus also in $S$. Craft now the formula $\textsc{Inj}(f_{A_0})$ for the particular $A_0$ received as input. By Proposition 4.6, for most $f_{A_0}$ there exists a certificate of injectivity $C_{A_0}$ such that $\textsc{Cert}(C_{A_0})$ is true and, in fact, has no free variables. Consider this certificate as a partial restriction and apply it to the implication above. Since $\mathbf{TC}^0$-Frege is closed under restrictions, there must be a polynomial-size proof of $\textsc{Inj}(f_{A_0})$, and so $S$ also proves this efficiently. Recall that, as noted in Remark 3.9, Impagliazzo's observation guarantees that under the existence of an automating algorithm we get constructive feasible interpolation, so from the proof of $\textsc{Inj}(f_{A_0})$ we can get a circuit that breaks one bit of the given output. By iterating this process we can recover the entire preimage. This procedure works as long as $f_{A_0}$ is injective and admits a certificate of injectivity, but by Proposition 4.6 this is the case with overwhelming probability. Then, by Lemma 4.7, we break LWE and get the desired conclusion. □

## Acknowledgments

The problem of looking at the quantum non-automatability of strong proof systems was suggested to us by three different people. We thank Vijay Ganesh for bringing it up during the Dagstuhl Seminar 22411 *Theory and Practice of SAT and Combinatorial Solving*. We thank Susanna F. de Rezende for bringing our attention to the problem later and for insightful comments and careful proofreading. We would also like to thank Ján Pich for independently pointing us to the problem and for discussing many details with us. We are particularly grateful for him directing us to the work of Soltys and Cook on LA, which greatly simplified our formalization. We also thank Rahul Santhanam for his insights and conversations and Yanyi Liu for pointing us to the existence of the certificates of injectivity. We also thank María Luisa Bonet, Ronald de Wolf, Alex Lombardi, Eli Goldin, Peter Hall, Angelos Pelecanos, Daniele Micciancio, Michael Soltys and Erfan Khaniki for useful comments and pointers, and Jonas Conneryd for careful proofreading of a draft of this text.

A preliminary version of this work was presented at the poster session of QIP 2024. We are thankful to the anonymous reviewers and their suggestions.

This work was done in part while the authors were visiting the Simons Institute for the Theory of Computing at UC Berkeley during the spring of 2023 for the *Meta-Complexity* and *Extended Reunion: Satisfiability* programs.

The first author was supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

## References


[AB04] A. Atserias and M. L. Bonet, "On the automatizability of resolution and related propositional proof systems," *Information and Computation*, vol. 189, no. 2, pp. 182–201, 2004.

[AB09] S. Arora and B. Barak, *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009.

[ABMP06] M. Alekhnovich, S. Buss, S. Moran, and T. Pitassi, "Minimum propositional proof length is **NP**-hard to linearly approximate," in *Mathematical Foundations of Computer Science 1998: 23rd International Symposium*, Springer, 2006, pp. 176–184.

[AD97] M. Ajtai and C. Dwork, "A public-key cryptosystem with worst-case/average-case equivalence," in *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, 1997, pp. 284–293.

[Ajt96] M. Ajtai, "Generating hard instances of lattice problems," in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, 1996, pp. 99–108.

[AM11] A. Atserias and E. Maneva, "Mean-payoff games and propositional proofs," *Information and Computation*, vol. 209, no. 4, pp. 664–691, 2011.

[AM20] A. Atserias and M. Müller, "Automating resolution is **NP**-hard," *Journal of the ACM (JACM)*, vol. 67, no. 5, pp. 1–17, 2020.

[AR05] D. Aharonov and O. Regev, "Lattice problems in $\mathbf{NP} \cap \mathbf{coNP}$," *Journal of the ACM (JACM)*, vol. 52, no. 5, pp. 749–765, 2005.

[AR08] M. Alekhnovich and A. A. Razborov, "Resolution is not automatizable unless $\mathbf{W}[\mathbf{P}]$ is tractable," *SIAM Journal on Computing*, vol. 38, no. 4, pp. 1347–1363, 2008.

[Ban93] W. Banaszczyk, "New bounds in some transference theorems in the geometry of numbers," *Mathematische Annalen*, vol. 296, pp. 625–635, 1993.

[BDG+04] M. L. Bonet, C. Domingo, R. Gavalda, A. Maciel, and T. Pitassi, "Non-automatizability of bounded-depth frege proofs," *computational complexity*, vol. 13, pp. 47–68, 2004.

[Bel20] Z. Bell, "Automating regular or ordered resolution is **NP**-hard," in *Electronic colloquium on computational complexity*, 2020.

[BN21] S. Buss and J. Nordström, "Proof complexity and SAT solving.," *Handbook of Satisfiability*, vol. 336, pp. 233–350, 2021.

[BP23] H. Bennett and C. Peikert, "Hardness of the (Approximate) Shortest Vector Problem: A Simple Proof via Reed-Solomon Codes," in *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2023)*, vol. 275, 2023, 37:1–37:20.

[BP96] P. Beame and T. Pitassi, "Simplified and improved resolution lower bounds," in *Proceedings of 37th Conference on Foundations of Computer Science*, 1996, pp. 274–282.

[BPR00] M. L. Bonet, T. Pitassi, and R. Raz, "On interpolation and automatization for frege systems," *SIAM Journal on Computing*, vol. 29, no. 6, pp. 1939–1967, 2000.

[BPT14] A. Beckmann, P. Pudlák, and N. Thapen, "Parity games and propositional proofs," *ACM Transactions on Computational Logic (TOCL)*, vol. 15, no. 2, pp. 1–30, 2014.

[CN10] S. Cook and P. Nguyen, *Logical Foundations of Proof Complexity*. Cambridge University Press, 2010.

[CR79] S. A. Cook and R. A. Reckhow, "The relative efficiency of propositional proof systems," *Logic, Automata, and Computational Complexity*, 1979.

[DMS11] S. Dantchev, B. Martin, and S. Szeider, "Parameterized proof complexity," *Computational Complexity*, vol. 20, pp. 51–85, 2011.

[dRez21] S. F. de Rezende, "Automating tree-like resolution in time $n^{o(n \log n)}$ is ETH-hard," *Procedia Computer Science*, vol. 195, pp. 152–162, 2021, Proceedings of the XI Latin and American Algorithms, Graphs and Optimization Symposium.

[dRGN+21] S. F. de Rezende, M. Göös, J. Nordström, T. Pitassi, R. Robere, and D. Sokolov, "Automating algebraic proof systems is **NP**-hard," in *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, 2021, pp. 209–222.

[Gar20] M. Garlík, "Failure of feasible disjunction property for $k$-DNF resolution and **NP**-hardness of automating it," *arXiv preprint arXiv:2003.10230*, 2020.

[GKMP20] M. Göös, S. Koroth, I. Mertz, and T. Pitassi, "Automating cutting planes is **NP**-hard," in *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, 2020, pp. 68–77.

[GKVW20] R. Goyal, V. Koppula, S. Vusirikala, and B. Waters, "On perfect correctness in (lockable) obfuscation," in *Theory of Cryptography Conference*, Springer, 2020, pp. 229–259.

[Gro96] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, 1996, pp. 212–219.

[HP11] L. Huang and T. Pitassi, "Automatizability and simple stochastic games," in *International Colloquium on Automata, Languages, and Programming*, Springer, 2011, pp. 605–617.

[IR22] D. Itsykson and A. Riazanov, "Automating OBDD proofs is **NP**-hard," in *47th International Symposium on Mathematical Foundations of Computer Science (MFCS 2022)*, 2022.

[Iwa97] K. Iwama, "Complexity of finding short resolution proofs," in *Mathematical Foundations of Computer Science 1997*, Springer Berlin Heidelberg, 1997, pp. 309–318.

[Jer05] E. Jerábek, "Weak pigeonhole principle, and randomized computation," Ph.D. dissertation, Faculty of Mathematics and Physics, Charles University, Prague, 2005.

[Jeř23] E. Jeřábek, "Elementary analytic functions in $\mathrm{VTC}^0$," *Annals of Pure and Applied Logic*, vol. 174, no. 6, p. 103 269, 2023.

[KP98] J. Krajíček and P. Pudlák, "Some consequences of cryptographical conjectures for $S^1_2$ and EF," *Information and Computation*, vol. 140, no. 1, pp. 82–94, 1998.

[Kra19] J. Krajíček, *Proof Complexity* (Encyclopedia of Mathematics and its Applications). Cambridge University Press, 2019.

[Kra22] J. Krajíček, "Information in propositional proofs and algorithmic proof search," *The Journal of Symbolic Logic*, vol. 87, no. 2, pp. 852–869, 2022.

[Kra95] J. Krajíček, *Bounded Arithmetic, Propositional Logic and Complexity Theory* (Encyclopedia of Mathematics and its Applications). Cambridge University Press, 1995.

[Kra97] J. Krajíček, "Interpolation theorems, lower bounds for proof systems, and independence results for bounded arithmetic," *The Journal of Symbolic Logic*, vol. 62, no. 2, pp. 457–486, 1997.

[Mic11] D. Micciancio, "The geometry of lattice cryptography," in *Foundations of Security Analysis and Design VI: FOSAD Tutorial Lectures*, A. Aldini and R. Gorrieri, Eds. Springer Berlin Heidelberg, 2011, pp. 185–210.

[MP12] D. Micciancio and C. Peikert, "Trapdoors for lattices: Simpler, tighter, faster, smaller," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Springer, 2012, pp. 700–718.

[MPW19] I. Mertz, T. Pitassi, and Y. Wei, "Short proofs are hard to find," in *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*, 2019.

[Pap23] T. Papamakarios, "Depth $d$ frege systems are not automatable unless $\mathbf{P} = \mathbf{NP}$," in *Electronic colloquium on computational complexity*, 2023.

[Pei+16] C. Peikert *et al.*, "A decade of lattice cryptography," *Foundations and Trends in Theoretical Computer Science*, vol. 10, no. 4, pp. 283–424, 2016.

[PS22] J. Pich and R. Santhanam, "Learning Algorithms Versus Automatability of Frege Systems," in *49th International Colloquium on Automata, Languages, and Programming (ICALP 2022)*, vol. 229, 2022, 101:1–101:20.

[Pud03] P. Pudlák, "On reducibility and symmetry of disjoint **NP** pairs," *Theoretical Computer Science*, vol. 295, no. 1, pp. 323–339, 2003, Mathematical Foundations of Computer Science.

[Pud09] P. Pudlák, "Quantum deduction rules," *Annals of Pure and Applied Logic*, vol. 157, no. 1, pp. 16–29, 2009.

[Pud97] P. Pudlák, "Lower bounds for resolution and cutting plane proofs and monotone computations," *The Journal of Symbolic Logic*, vol. 62, no. 3, pp. 981–998, 1997.

[PW08] C. Peikert and B. Waters, "Lossy trapdoor functions and their applications," in *Proceedings of the fortieth annual ACM symposium on Theory of computing*, 2008, pp. 187–196.

[Reg09] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," *Journal of the ACM (JACM)*, vol. 56, no. 6, pp. 1–40, 2009.

[RS21] N. Rahman and V. Shpilrain, *MOBS (Matrices Over Bit strings) public key exchange*, 2021. arXiv: 2106.01116.

[SC04] M. Soltys and S. Cook, "The proof complexity of linear algebra," *Annals of Pure and Applied Logic*, vol. 130, no. 1, pp. 277–323, 2004, Papers presented at the 2002 IEEE Symposium on Logic in Computer Science (LICS).

[Sho94] P. Shor, "Algorithms for quantum computation: Discrete logarithms and factoring," in *Proceedings 35th Annual Symposium on Foundations of Computer Science*, 1994, pp. 124–134.

[Yao93] A. C.-C. Yao, "Quantum circuit complexity," in *Proceedings of 1993 IEEE 34th Annual Foundations of Computer Science*, 1993, pp. 352–361.

# A The theories LA and $\mathsf{LA}_{\mathbb{Q}}$

Appendix A.1 below lists the axioms of the theory LA of linear algebra of [SC04], together with several theorems proven inside the theory in the original paper. Appendix A.2 proves that the conservative extension $\mathsf{LA}_{\mathbb{Q}}$ admits a propositional translation into $\mathbf{TC}^0$-Frege.

## A.1 Axioms and basic theorems of LA

1. **Equality axioms**
   (a) $x = x$
   (b) $x = y \to y = x$
   (c) $(x = y \wedge y = z) \to x = z$
   (d) $\bigwedge_i^n (x_i = y_i) \to f(\bar{x}) = f(\bar{y})$
   (e) $i_1 = j_1, i_2 = j_2, i_1 \leq i_2 \to j_1 \leq j_2$.
2. **Axioms for indices**
   (a) $i + 0 = i$
   (b) $i + (j + 1) = (i + j) + 1$
   (c) $i \cdot (j + 1) = (i \cdot j) + i$
   (d) $i + 1 = j + 1 \to i = j$
   (e) $i + 1 \neq 0$
   (f) $i \leq i + j$
   (g) $i \leq j, j \leq i$
   (h) $i \leq j, i + k = j \to (j - i = k)$
   (i) $i \leq j, i + k = j \to (i \neq j \to j - i = 0)$
   (j) $j \neq 0 \to \mathsf{rem}(i, j) < j$
   (k) $j \neq 0 \to i = j \cdot \mathsf{div}(i, j) + \mathsf{rem}(i, j)$
   (l) $\alpha \to \mathsf{cond}(\alpha, i, j) = i$
   (m) $\neg\alpha \to \mathsf{cond}(\alpha, i, j) = j$
3. **Axioms for field elements**
   (a) $0 \neq 1 \wedge a + 0 = a$
   (b) $a + (-a) = 0$
   (c) $1 \cdot a = a$
   (d) $a \neq 0 \to a \cdot (a^{-1}) = 1$
   (e) $a + b = b + a$
   (f) $a \cdot b = b \cdot a$
   (g) $a + (b + c) = (a + b) + c$
   (h) $a \cdot (b \cdot c) = (a \cdot b) \cdot c$
   (i) $a \cdot (b + c) = a \cdot b + a \cdot c$
   (j) $\alpha \to \mathsf{cond}(\alpha, a, b) = a$
   (k) $\neg\alpha \to \mathsf{cond}(\alpha, a, b) = b$
4. **Axioms for matrices**
   (a) $(i = 0 \vee \mathsf{r}(A) < i \vee j = 0 \vee \mathsf{c}(A) < j) \to \mathsf{e}(A, i, j) = 0$
   (b) $\mathsf{r}(A) = 1, \mathsf{c}(A) = 1 \to \Sigma(A) = \mathsf{e}(A, 1, 1)$
   (c) $\mathsf{c}(A) = 1 \to \sigma(A) = \sigma(A^\intercal)$
   (d) $\mathsf{r}(A) = 0 \vee \mathsf{c}(A) = 0 \to \Sigma(A) = 0$
5. **Theorems for Ring Properties**
   (a) $\max(i, j) = \max(j, i)$
   (b) $\max(i, \max(j, k)) = \max(\max(i, j), k)$
   (c) $\max(i, \max(j, k)) = \max(\max(i, j), \max(i, k))$
   (d) $A + 0 = A$
   (e) $A + (-1)A = 0$
   (f) $AI = A$ and $IA = A$
   (g) $A + B = B + A$
   (h) $A + (B + C) = (A + B) + C$
   (i) $A(BC) = (AB)C$
   (j) $A(B + C) = AB + CA$
   (k) $(B + C)A = BA + CA$
   (l) $\Sigma 0 = 0_{\text{field}}$
   (m) $\Sigma(cA) = c\Sigma(A)$
   (n) $\Sigma(A + B) = \Sigma(A) + \Sigma(B)$
   (o) $\Sigma(A) = \Sigma(A^\intercal)$
6. **Theorems for Module Properties**
   (a) $(a + b)A = aA + bA$
   (b) $a(A + B) = aA + aB$
   (c) $(ab)A = a(bA)$
7. **Theorems for Inner Product**
   (a) $A \cdot B = B \cdot A$
   (b) $A \cdot (B + C) = A \cdot B + A \cdot C$
   (c) $aA \cdot B = a(A \cdot B)$
8. **Miscellaneous Theorems**
   (a) $a(AB) = (aA)B \wedge (aA)B = A(aB)$
   (b) $(AB)^\intercal = B^\intercal A^\intercal$
   (c) $I^\intercal = I$
   (d) $0^\intercal = 0$
   (e) $(A^\intercal)^\intercal = A$

## A.2 The propositional translation for $\mathsf{LA}_{\mathbb{Q}}$

We show that theorems of $\mathsf{LA}_{\mathbb{Q}}$ still have short propositional proofs in $\mathbf{TC}^0$-Frege despite the presence of new symbols and axioms not present in LA.

The first step in the propositional translation is the conversion of LA formulas into propositional ones. The translation is analogous to the usual propositional translations used elsewhere in bounded arithmetic (see, for example, [CN10; Kra95]). Let $\varphi$ be a formula of $\mathsf{LA}_{\mathbb{Q}}$ and let $\sigma$ be an object assignment that assigns a natural number to each free index variable occurring in $\varphi$ and to each term of the form $\mathsf{r}(A)$ and $\mathsf{c}(A)$ occurring in $\varphi$. We denote by $N$ the maximum value in the range of $\sigma$. For every variable $q$ standing for a rational number in $\varphi$, we introduce enough Boolean variables to represent $q$ as a fraction $a/b$, where $a$ and $b$ are integers represented in binary. We may assume that $N$ is also an upper bound on the binary precision of these integers. We adopt the convention that denominators are always positive. Note that the number of Boolean variables introduced is at most polynomial in $N$.

The translation of $\varphi$ then proceeds by substituting every function and predicate symbol by the corresponding $\mathbf{TC}^0$ circuit of the appropriate size, which is also at most $\mathrm{poly}(N)$. It is easy to verify that all the functions and predicate symbols in LA are computable in $\mathbf{TC}^0$, and this is also the case for the extended vocabulary ($<_{\mathbb{Q}}$ and $\mathsf{int}$). For $<_{\mathbb{Q}}$, given $a/b$ and $c/d$, we check whether $ac < bd$, which only requires operations over the integers. For $\mathsf{int}$ we shall use the circuit computing whether the rational $q = a/b$ satisfies that $\mathrm{MOD}(a, b) = 0$. This only requires the standard remainder function, computable in $\mathbf{TC}^0$, plus an equality check. We denote by $||\varphi||_\sigma$ the propositional formula obtained by carrying out this translation process. The size of $||\varphi||_\sigma$ is again polynomial in $N$.

Now, given an LA proof $\pi$ of a sentence $\varphi$ and an object assignment $\sigma$, we translate $\pi$ into a $\mathbf{TC}^0$-Frege proof of $||\varphi||_\sigma$. It suffices to translate each line in the proof into its corresponding propositional formula and observe that the required inference steps can be carried out in $\mathbf{TC}^0$-Frege. More precisely, the underlying proof system for LA and $\mathsf{LA}_{\mathbb{Q}}$ is the sequent calculus, and we may think of $\mathbf{TC}^0$-Frege in its sequent calculus formulation, PTK. Since LA and $\mathsf{LA}_{\mathbb{Q}}$ formulas are quantifier-free, no derivation rules for quantifiers are present in $\pi$. Every inference step of an LA proof matches the corresponding sequent calculus rule of the propositional sequent calculus. It is also not hard to see that the cut rule is always over a $\mathbf{TC}^0$ circuit, since the $\mathsf{LA}_{\mathbb{Q}}$ formula over which we cut translates into a $\mathbf{TC}^0$ circuit.

The only problem occurs when reaching a leaf in the proof $\pi$, which corresponds to an axiom of LA or $\mathsf{LA}_{\mathbb{Q}}$. These are not axioms of $\mathbf{TC}^0$-Frege and hence require a proof to be appended in the translation. Cook and Soltys observed that when instantiated over the rationals, all of the axioms of LA are either directly proven or follow easily from the basic properties of arithmetic proven in [BPR00] inside $\mathbf{TC}^0$-Frege. Hence, the only thing left to complete the propositional translation for $\mathsf{LA}_{\mathbb{Q}}$ is to provide small $\mathbf{TC}^0$-Frege proofs of the new axioms not present in LA.

**Lemma A.1.** *There are polynomial-size $\mathbf{TC}^0$-Frege proofs of the propositional translation of the axioms of $\mathsf{LA}_{\mathbb{Q}}$.*

*Proof.* The axioms of LA were handled in the original work of Cook and Soltys (see Theorem 6.3 in [SC04]). Furthermore, the axioms imposing that $<_{\mathbb{Q}}$ is an ordering relation were already proven in [BPR00] as well (these are precisely the lemmas proven in their Section 7.2). We therefore focus on the translation of the axioms for $\mathsf{int}$.

For the sake of consistency with the previous work of Bonet, Pitassi, and Raz we adopt here the notation $[a]_b$ for the $\mathrm{MOD}(a, b)$ function and $\mathrm{div}_b(a)$ for the integer division between $a$ and $b$. We also reuse the following lemmas proved by them inside $\mathbf{TC}^0$-Frege, where L7._ stands for the corresponding lemma in [BPR00]:

(L7.19) $(a < b) \vee (b < a) \vee (a = b)$.

(L7.27) $a = [a]_b + \mathrm{div}_b(a) \cdot b$.

(L7.28) $x + y \cdot p = u + v \cdot p \wedge y < v \rightarrow p \leq x$.

(L7.29) $[a]_b = [a + k \cdot b]_b$.

We are now ready to write the proofs for the axioms ($\mathrm{Int}_1$), ($\mathrm{Int}_2$), and ($\mathrm{Int}_3$).

($\mathrm{Int}_1$) Let $x$ and $y$ be represented by the fractions $a/b$ and $c/d$ respectively. The translation of the axiom

$$\mathsf{int}(x) \wedge \mathsf{int}(y) \rightarrow \mathsf{int}(x + y)$$

yields the propositional formula

$$[a]_b = 0 \wedge [c]_d = 0 \longrightarrow [ad + bc]_{bd} = 0.$$

From $[a]_b = 0$ and $[c]_d = 0$, L7.27 gives us that $a = \mathrm{div}_b(a) \cdot b$ and $c = \mathrm{div}_d(c) \cdot d$. Then,

$$\begin{aligned}
[ad + bc]_{bd} &= [\underbrace{\mathrm{div}_b(a) \cdot b}_{a} \cdot d + \underbrace{\mathrm{div}_d(c) \cdot d}_{c} \cdot b]_{bd} \\
&= [bd \cdot (\mathrm{div}_b(a) + \mathrm{div}_d(c))]_{bd} \\
&= [0]_{bd} \\
&= 0
\end{aligned}$$

where the second to last equality follows by applying L7.29.

($\mathrm{Int}_2$) In this case the translation of

$$\mathsf{int}(x) \wedge \mathsf{int}(y) \longrightarrow \mathsf{int}(x \cdot y)$$

yields the formula

$$[a]_b = 0 \wedge [c]_d = 0 \longrightarrow [ac]_{bd} = 0.$$

We have again that L7.27 gives us that $a = \mathrm{div}_b(a) \cdot b$ and $c = \mathrm{div}_d(c) \cdot d$. Then,

$$\begin{aligned}
[ac]_{bd} &= [ac + (-\mathrm{div}_b(a) \cdot \mathrm{div}_d(c)) \cdot bd]_{bd} \\
&= [ac - \underbrace{\mathrm{div}_b(a) \cdot b}_{a} \cdot \underbrace{\mathrm{div}_d(c) \cdot d}_{c}]_{bd} \\
&= [ac - ac]_{bd} \\
&= [0]_{bd} \\
&= 0
\end{aligned}$$

where the first equality follows again from L7.29.

($\mathrm{Int}_3$) We first write the propositional translation of

$$\mathsf{int}(x) \wedge 0 < x \longrightarrow 1 \leq x.$$

Recall that we adopted the convention that denominators of fractions are always positive, and the comparator circuit between two rationals $a/b$ and $c/d$ checks the integer inequality $ac < bd$. The consequent $1 \leq x$ stands for $1 = x \vee 1 < x$, which translates as $\mathrm{div}_b(a) = 1 \vee b < a$ when writing $x$ and $a/b$. Thus, the formula to prove is

$$[a]_b = 0 \wedge 0 < a \longrightarrow \mathrm{div}_b(a) = 1 \vee b < a.$$

By L7.19, either $b < a$, $a < b$ or $a = b$. If $b < a$, we are done. If $a = b$, using L7.27, it is easy to show that $\mathrm{div}_a(a) = 1$, since by L7.29,

$$[a]_a = [0 + a]_a = [0]_a = 0$$

and thus

$$a = \mathrm{div}_a(a) \cdot a + [a]_a = \mathrm{div}_a(a) \cdot a.$$

Since by assumption $0 < a$, we have $a \neq 0$, so $\mathrm{div}_a(a) = 1$. Then,

$$\mathrm{div}_b(a) = \mathrm{div}_a(a) = 1.$$

Finally, if $a < b$, we prove that the antecedent of the formula is falsified. We first show that $\mathrm{div}_a(b) = 0$. Suppose not, then it must be $\mathrm{div}_a(b) > 0$. When taking $x = a$, $y = 0$, $p = b$ and $v = \mathrm{div}_b(a)$ in L7.28 above, we immediately get $b \leq a$, contradicting $a < b$.

Now that we have $\mathrm{div}_b(a) = 0$, by L7.27 we get $a = [a]_b$. But if both $[a]_b = 0$ and $0 < a$, we get a contradiction. □

**Theorem A.2** (Propositional translation for $\mathsf{LA}_{\mathbb{Q}}$)**.** *For every theorem $\varphi$ of $\mathsf{LA}_{\mathbb{Q}}$ and every object assignment $\sigma$, the propositional formula $||\varphi||_\sigma$ admits polynomial-size $\mathbf{TC}^0$-Frege proofs.*

*Proof.* The proof is analogous to Theorem 6.3 in [SC04], except we need to handle the new axioms. By Lemma A.1 above, the translations of the new axioms have short $\mathbf{TC}^0$-Frege proofs. This completes the proof. □

# B Proof of Proposition 3.4

We prove the equivalence between the machine-based and circuit-based definitions in the three settings.

(i) **Classical automatability.** For the forward direction, suppose $A$ is an automating deterministic Turing machine. In order to simulate $A$ by a circuit, we need to introduce a uniform bound on the running time of $A$. We know $A$ runs in time $\mathsf{size}_S(\varphi)^c$ for some constant $c$. Consider now the machine $A'$ that takes as input both $\varphi$ and a size parameter $s$ in unary and runs $A(\varphi)$ for $s^c$ steps, and outputs a proof if one was found, and some other string otherwise. This machine $A'$ can be simulated by a uniform circuit family of size $O((|\varphi| + s)^{2c})$, which is still polynomial in $|\varphi| + s$, and which outputs a proof of size polynomial in $s$ if one exists.

For the backwards direction, assuming a circuit family $\{C_{n,s}\}_{n,s\in\mathbb{N}}$, the machine on input $\varphi$ simulates $C_{|\varphi|,1}(\varphi)$, $C_{|\varphi|,2}(\varphi)$ and so on, checking every time whether the output proof is valid, up to the first value of $s$ for which a valid proof is obtained. This takes time polynomial in $\mathsf{size}_S(\varphi)$.

(ii) **Randomized automatability.** The argument here is similar, except that we have to account for the equivalence between the bounded expected running time of the machine and the bounded error probability of the circuits.

For the forward direction, let $R$ be a probabilistic machine automating $S$ in expected time $\mathsf{size}_S(\varphi)^c$ for some constant $c$. Let $T_\varphi$ be the random variable that denotes the number of steps $R$ takes to find a proof on input $\varphi$, when $\varphi$ does have some proof. We know that $\mathrm{E}[T_\varphi] \leq \mathsf{size}_S(\varphi)^c$. Consider now the modified machine $R'$ that takes $\varphi$ and a size parameter $s$ and simulates $R(\varphi)$ for $k \cdot (n+s)^c$ steps, for some constant $k$ such as $k = 100$. This machine can be turned into a random circuit with $k \cdot (n+s)^c$ random bits. It just suffices to argue that for at least $2/3$ of the choices for the random bits, the circuit will output a proof when one exists. Indeed, by Markov's inequality, the probability that $R'$ might not output a proof in time $k \cdot (n+s)^c$ is just $\Pr\left[T_\varphi > k \cdot \mathrm{E}[T_\varphi]\right] \leq 1/k$, which bounds the error of the circuit as desired.

For the backwards direction, from the sequence $\{C_{n,s}\}_{n,s\in\mathbb{N}}$ of randomized circuits we get an error-bounded probabilistic Turing machine $R(\varphi, s)$ that first obtains the description of $C_{|\varphi|,s}$ (recall that the circuit family is uniform) and then simulates $C_{|\varphi|,s}(\varphi)$. This machine $R$ always halts after $(|\varphi| + s)^{O(1)}$ steps, and, whenever a proof of size $s^c$ exists, finds one with probability at least $2/3$. Now, consider the machine $R'$ that takes as input just the formula $\varphi$ and runs $R(\varphi, 1), R(\varphi, 2), \ldots$ and so on, until a proof is found. For very small values of $s$ the the machine $R$ will never find a proof, because none exists. Once we get to values of $s$ large enough such that $s^c \geq \mathsf{size}_S(\varphi)$, we might still be unlucky and not find a proof when running $R(\varphi, s)$, and move to $R(\varphi, s+1)$. Note, however, that the number of times we may increments the parameter $s$ before a proof is found follows a geometric distribution, and so the expected number of trials is at most $1/p$, where $p$ is the probability of success. Since $p$ is at least $2/3$, the expected number of times we will increment $s$ before a proof is found is at most $3/2$. Altogether, the machine $R'$ will run in expected time polynomial in $\mathsf{size}_S(\varphi)$.

(iii) **Quantum automatability.** The proof is identical to (ii). By Yao's result that quantum circuits can simulate quantum Turing machines running in time $T$ in size $O(T^2)$ [Yao93], we get the right transformations between circuits and machines, and the probability analysis is exactly the same.

□

# C Properties of random lattices (Proof of Lemma 2.4)

This section proves the two statements of Lemma 2.4. We start by proving that almost every randomly selected matrix is full-rank (Lemma 2.4.i). We then prove two technical lemmas. Finally we show that almost every randomly selected full rank matrix generates a lattice with no short vectors (Lemma 2.4.ii).

From now on, unless otherwise specified, we consider lattices of the form $\mathcal{L}_q(A)$ where $A \in \mathbb{Z}_q^{m\times n}$ and $\text{rank}(A) = n$. Note that for any such lattice $|\mathcal{L}_q(A)| = q^n$.

*Proof of Lemma 2.4.i.* When selecting a column vector there are $q^m$ different options. At each step $i$ the previously selected columns span a subspace of $\mathbb{Z}_q^m$ with $q^{i-1}$ elements, meaning that on the $i$-th selection the odds of selecting a linearly dependent vector are only $q^{i-1}/q^m = 1/q^{m-i+1}$. For each step $i$, this probability is less than $1/q^{m-n+1}$. By union bounding over the $n$ opportunities, the probability of ever selecting a linearly dependent column is less than the sum of these probabilities which in turn is less than $n/q^{m-n+1}$. □

**Lemma C.1.** *Let $\{\mathcal{L}_q\}$ be the set of all the possible distinct rank-n lattices in $\mathbb{Z}_q^m$. It holds that*

$$|\{\mathcal{L}_q\}| = \prod_{i=0}^{n-1}\left(\frac{q^m - q^i}{q^n - q^i}\right).$$

*Proof.* The cardinality of $\{\mathcal{L}_q\}$ is equal to the number of rank $n$ bases divided by the number of possible bases for each given lattice. Formally,

$$|\{\mathcal{L}_q\}| = \frac{|\{A|\ \text{rank}(A) = n\}|}{|\{A'|\mathcal{L}_q(A) = \mathcal{L}_q(A')\}|}.$$

We take an algorithmic approach to counting the number of $A$ for which $\text{rank}(A) = n$. To select such an $A$, we first set $a_0$ equal to one of the $q^m - 1$ non-zero points in $\mathbb{Z}_q^m$. Then for each subsequent $i$ we set $a_i$ equal to a point in $\mathbb{Z}_q^m$ not contained in the rank $i$ lattice spanned by $(a_0, \ldots, a_{i-1})$. We know that there are $q^i$ vectors in that lattice leaving us with $q^m - q^i$ possible choices for $a_i$. To avoid double counting the permutations of a given basis we divide by $n!$, concluding that there are $\prod_{i=0}^{n-1}(q^m - q^i)/n!$ matrices $A$ with $\text{rank}(A) = n$.

Next, to count the number of possible bases $A'$ we first note that any set of $n$ linearly independent vectors in $\mathcal{L}_q(A)$ is a basis of $\mathcal{L}_q(A)$. Then we follow the same method as above for generating a basis except our choices are now limited to the $q^n$ vectors in the lattice. So we end up with a total number of possible bases of $\prod_{i=0}^{n-1}(q^n - q^i)/n!$.

If we divide the number of rank-$n$ bases by the number of bases per lattice we get $\prod_{i=0}^{n-1}\left(\frac{q^m-q^i}{q^n-q^i}\right)$. □

**Lemma C.2.** *When $q \geq n \geq 1$, $\log_q(q+1)(n-1) \leq n$.*

*Proof.* Because $\log_q(q+1)$ is monotonically decreasing it suffices to show that $\log_n(n+1)(n-1) \leq n$. By change of basis and reordering this is equivalent to proving that

$$\frac{\ln(n+1)}{\ln(n)} \leq \frac{n}{n-1}.$$

It is well known that

$$\frac{d(\ln(n))}{dn} = \frac{1}{n},$$

which is also monotonically decreasing, meaning that

$$\ln(n+1) \leq \ln(n) + 1\frac{1}{n}.$$

Thus,

$$\frac{\ln(n+1)}{\ln(n)} \leq \frac{\ln(n) + \frac{1}{n}}{\ln(n)} = 1 + \frac{1}{n\ln(n)} \leq 1 + \frac{1}{n-1} = \frac{n}{n-1}.$$

□

*Proof of Lemma 2.4.ii.* Every lattice with a short vector can be specified by a tuple of a lattice of rank $n-1$ and a short vector. All vectors with length less than $r$ must lie in the region $(-r, r)^m$ so we know there are fewer than $(2r+1)^m$ short vectors. Combining this with the number of lattices of rank $n-1$ from Lemma C.1 we get that

$$|\{\mathcal{L}_q(A) \mid \lambda_1(\mathcal{L}_q(A)) \leq r\}| \leq (2r+1)^m \prod_{i=0}^{n-2}\left(\frac{q^m - q^i}{q^{n-1} - q^i}\right).$$

If we divide this upper bound on the number of lattices with short vectors by the exact count of the number of total lattices from Lemma C.1 we can see that the fraction of lattices which contain a vector of length less than $r$ is less than

$$\begin{aligned}
\frac{(2r+1)^m \prod_{i=0}^{n-2}\left(\frac{q^m-q^i}{q^{n-1}-q^i}\right)}{\prod_{i=0}^{n-1}\left(\frac{q^m-q^i}{q^n-q^i}\right)} &= \frac{(2r+1)^m}{\frac{q^m-q^{n-1}}{q^n-q^{n-1}}} \prod_{i=0}^{n-2} \frac{\left(\frac{q^m-q^i}{q^{n-1}-q^i}\right)}{\left(\frac{q^m-q^i}{q^n-q^i}\right)} \\
&= (2r+1)^m \frac{q^n-q^{n-1}}{q^m-q^{n-1}} \prod_{i=0}^{n-2} \frac{q^n-q^i}{q^{n-1}-q^i} \\
&\leq (2r+1)^m \frac{q^n}{q^{m-1}} \left(\frac{q^n-q^{n-2}}{q^{n-1}-q^{n-2}}\right)^{n-1} \\
&= \frac{(2r+1)^m}{q^{m-n-1}} (q+1)^{(n-1)} \\
&= \frac{(2r+1)^m}{q^{m-n-1}} q^{\log_q(q+1)(n-1)} \\
&\leq \frac{(2r+1)^m}{q^{m-2n-1}}. && \text{(by Lemma C.2)}
\end{aligned}$$

□