Polynomial-Time Quantum Algorithms for Pell's Equation and the Principal Ideal Problem[*]

Sean Hallgren[†]
NEC Laboratories America, Inc.
4 Independence Way
Princeton, NJ 08540
hallgren@nec-labs.com

May 9, 2006

Abstract

We give polynomial-time quantum algorithms for three problems from computational algebraic number theory. The first is Pell's equation. Given a positive non-square integer d, Pell's equation is $x^2 - dy^2 = 1$ and the goal is to find its integer solutions. Factoring integers reduces to finding integer solutions of Pell's equation, but a reduction in the other direction is not known and appears more difficult. The second problem we solve is the principal ideal problem in real quadratic number fields. This problem, which is at least as hard as solving Pell's equation, is the one-way function underlying the Buchmann-Williams key exchange system, which is therefore broken by our quantum algorithm. Finally, assuming the generalized Riemann hypothesis, this algorithm can be used to compute the class group of a real quadratic number field.

1 Introduction

The search for quantum algorithms is a fundamental goal of quantum computing, and there is a great deal of interest in how quantum computing changes the tractable-intractable boundary in polynomial-time computation. Shor's algorithms [Sho97] for factoring and discrete log resulted in much excitement over the potential of quantum computation. At

[*] A preliminary version of this paper appeared in STOC'02 [Hal02]. Work done at MSRI, U.C. Berkeley, and Caltech.

[†] Supported at Caltech in part by an NSF Mathematical Sciences Postdoctoral Fellowship, NSF through Caltech's Institute for Quantum Information, NSF under grant no. 0049092 (previously 9876172) and The Charles Lee Powell Foundation. Part of this work done while the author was at MSRI and U.C. Berkeley, with partial support from DARPA QUIST Agreement No. F30602-01-2-0524.

the same time it also showed that current methods of cryptography used on the Internet will have to be changed if quantum computers can be built. Since Shor's algorithms, much work has gone into a generalized problem called the hidden subgroup problem [EHK99, HRTS00, GSVV01, FIM$^+$03, Kup03, MRRS04]. This problem includes as special cases previously solved problems such as factoring and discrete log, as well as still unsolved problems such as graph isomorphism. While progress has been made on the hidden subgroup problem, finding more problems where quantum computation has a significant advantage over classical computation has been difficult [Sho03].

In this paper, we give polynomial-time quantum algorithms for Pell's equation and the principal ideal problem. Using these algorithms, we are also able to compute the class group of a real quadratic number field, and break, when given a quantum computer, the key exchange protocol proposed by Buchmann and Williams [BW89a]. Besides giving natural, well-studied examples, this paper extends the hidden subgroup framework and points towards how to extend it further. This provides insights into the nature of Fourier sampling, the main workhorse of quantum algorithms. In particular, while the hidden subgroup problem can be defined over any group, Fourier sampling can only be performed over finite groups. One can view Shor's work as showing how to effectively use Fourier sampling when then underlying group is finitely generated. In this work we extend Fourier sampling to non-finitely generated groups, as there will be an underlying periodic function over the reals whose period we wish to approximate.

We also shed light on what cryptography might look like if quantum computers can be built. It is known that the current systems which assume factoring is computationally hard will be broken by quantum computers. Here we show that a potentially much more secure system can be broken too.

Pell's equation is one of the oldest problems studied in number theory. Given a positive non-square integer d, Pell's equation is $x^2 - dy^2 = 1$, and the goal is to find all integer solutions. The original algorithm for solving it is the second oldest number theory algorithm after Euclid's algorithm. The algorithm is due to Indian mathematicians around 1000 a.d. In 1768 Lagrange showed that there are an infinite number of solutions of the equation, and the following theorem is well-known. If (x_1, y_1) is the least positive solution of $x^2 - dy^2 = 1$, ordered by the value of $x_1 + y_1\sqrt{d}$, where d is a positive non-square integer, then all positive solutions are given by (x_n, y_n) for $n = 1, 2, 3, \ldots$, where $x_n + y_n\sqrt{d} = (x_1 + y_1\sqrt{d})^n$ [NZM91].

It is not actually possible to have a polynomial-time algorithm for finding the least positive solution (x_1, y_1), because it may take exponentially many bits to represent (the input size is $\log d$). Instead, the integer closest to the *regulator*, defined as $R = \ln(x_1 + y_1\sqrt{d})$, is computed. Given this integer $\lfloor R \rceil$ it is possible to compute R to arbitrary precision, and $\lfloor R \rceil$ uniquely identifies (x_1, y_1). In fact, it is enough to solve a closely related problem, which is finding the regulator of the ring $\mathbb{Z}[\sqrt{d}]$. This is the problem we solve, described precisely in the next section. We also solve another problem related to the ring $\mathbb{Z}[\sqrt{d}]$, called the *principal ideal problem.* The principal ideal problem is the following: given an invertible ideal I determine if there exists an $\alpha \in \mathbb{Q}(\sqrt{d})$ such that $I = \alpha\mathbb{Z}[\sqrt{d}]$, and if there is, find α. As with Pell's equation this value may be too large, but finding the closest integer to $\log \alpha$ is enough to uniquely identify α. Using our quantum algorithm for this problem and assuming the GRH, we can also compute the class group of a real quadratic number field. Full definitions will be given in the next section. For more about Pell's equation, see [Wil00, Len02].

The expected running time of the classical algorithms for these problems is measured using the function $L(a,b) = \exp(bn^a(\log n)^{1-a})$, where n is the input size. The goal is to reduce a to zero, which would be polynomial-time. The best algorithm for factoring integers has expected time $L(\frac{1}{3}, b)$ for some constant b [LL93]. Assuming the GRH, the best algorithms for Pell's equation and the principal ideal problem have expected time $L(\frac{1}{2}, b')$ [Buc89, Vol00], for some constant b', so there is an sub-exponential gap between the best known classical algorithms. Both the running time and the correctness of the classical algorithm for Pell's equation are based on the generalized Riemann hypothesis (GRH). The GRH appears in many algorithms in number theory and is used sometimes to show correctness and sometimes to analyze the running time. For example, the best classical algorithm for Pell's equation without assumptions is $O(d^{1/4}\text{polylog } d)$. In addition, while finding multiples of the regulator is in NP, finding the regulator itself is only known to be in NP under the GRH [BW89a]. Our quantum algorithms for computing the regulator (solving Pell's equation) and the principal ideal problem do not use any assumptions.

There are reductions from factoring to solving Pell's equation, and from solving Pell's equation to solving the principal ideal problem [BW89b]. However, Pell's equation and the principal ideal problem appear to be harder than factoring, and there are no reductions known in the other direction. This is reflected in the gap between the running times of the best algorithms for factoring and Pell's equation. Indeed, a key exchange system based on the principal ideal problem is proposed in [BW89b], and one of the motivations is that even if there turns out to be a polynomial-time algorithm for factoring, their system might still be unbreakable. Our quantum algorithm for the principal ideal problem breaks this key exchange system in this context.

Classically, computing the class group appears to be computationally tied to computing the regulator [Coh93], or in other words, there is no reason to compute one without the other. Curiously, this does not appear to be the case with respect to quantum algorithms. The primary problem we must overcome to compute the class group is the fact that group elements do not have unique representatives. Furthermore, the structure of the representatives is arbitrary, making techniques in [Wat01] inapplicable. To deal with this problem our algorithm must first compute the regulator, and then use the principal ideal problem algorithm as a subroutine to create superpositions over equivalence classes.

The quantum step in our algorithm for Pell's equation is a new procedure to efficiently approximate the period of a periodic function with irrational period. The algorithm for the principal ideal problem reduces to a discrete log type problem, but there is no longer an underlying group. Instead, a group-like subset of the reals modulo an irrational number is used. This prevents direct application of Shor's algorithms. Dealing with these problems, and the one for the class group, are the main technical pieces of the paper.

2 Background

We will use the following notation: a, b, c, d, i, j, k, l, m, n, p, q, N are non-negative integers, x, y, z, R, S are real numbers, and f, g, h are functions. When computing with real numbers we mean computing with approximations that are good enough for our purposes.

2.1 Quantum Computing Background

All problems that have quantum algorithms with superpolynomial or exponential speedups over the best known classical algorithm have quantum algorithms that use Fourier sampling [BV97]. Given a quantum state, *Fourier sampling* is the process of computing the Fourier transform and measuring the result. This primitive differs in two important ways from just computing the classical Fourier transform of a vector. First, the quantum Fourier transform can be computed exponentially faster. In the current context this means it can be computed in polynomial-time even though the state vector has exponential size. Second, the output of the primitive is not a vector of complex numbers, but only a classical bit string. A string is measured with probability equal to its amplitude squared. Thus, the phases are ignored, and there is far more restricted access to the output vector from the Fourier transform than in the classical case. Except for the problems solved in [vDHI03], it is sufficient to use the stronger primitive of Fourier sampling a function (for example [Sim97, Sho97, HRTS00, GSVV01, IMS01]).

Given a function f, *Fourier sampling the function f* is the process of creating the superposition $\frac{1}{\sqrt{|G|}}\sum_{g\in G}|g,f(g)\rangle$, optionally measuring the second register containing $f(g)$ (or just creating $\frac{1}{\sqrt{|G|}}\sum_{g\in G}f(g)|g\rangle$ in the case of [BV97]), and then Fourier sampling.

The most common application of Fourier sampling is the Hidden Subgroup Problem. In this problem a group G and a function $f : G \to S$ are given with the condition that there is an unknown subgroup $H \leq G$ such that the function f is constant on (left) cosets of H, and takes different values on different cosets. The task is to find a set of generators for H. The range of f is a set S. Fourier sampling the function f in this case first creates a state that is uniform on a coset of the subgroup $H \leq G$: $\frac{1}{\sqrt{|H|}}\sum_{h\in H}|g+h\rangle$, where g is chosen uniformly from G and $f(g)$ was the value measured in the second register. The second step computes the Fourier transform over G and measures. In this approach the hidden subgroup function is only used to set up a superposition which is essentially a characteristic function of a random coset.

For example, for the hidden subgroup problem over the finite cyclic group $\mathbb{Z}_{rm} = \{0,1,2,\ldots,rm-1\}$ with hidden subgroup $\mathbb{Z}_r = \{0,m,2m,\ldots,(r-1)m\}$, the Fourier transform "inverts" the period, and the resulting distribution is uniform over the points of $\mathbb{Z}_m = \{0,r,2r,\ldots,(m-1)r\}$. Sampling from this distribution allows one to compute r, and from r, the generator m. Shor's algorithm extends this idea to the infinite group $\mathbb{Z}$, by Fourier sampling f over a large enough cyclic subgroup $\mathbb{Z}_q$. In that case, for a subgroup $H = \langle m\rangle$, samples will be concentrated around integer multiples of q/m. In this paper we extend this further to the case of $G = \mathbb{R}$, where H can be generated by an irrational number.

One of the properties that makes Fourier sampling useful in the hidden subgroup problem is that the resulting distribution is independent of the coset. This can be seen from the convolution-multiplication property of the Fourier transform. Fourier sampling $\sum_{s\in S}c_S|s\rangle$, where S is a subset of some abelian group with addition and $c_S = \frac{1}{\sqrt{|S|}}$, is the same as Fourier sampling $\sum_{s\in S}c_S|k+s\rangle$, where k is a group element. We see that $F(\sum_s c_S|k+$

$s\rangle) = F(|k\rangle * \sum_s c_S|s\rangle) = (F|k\rangle) \cdot (F\sum_s c_S|s\rangle)$, where $*$ is convolution and $\cdot$ is point-wise multiplication. Here $F|k\rangle$ is the Fourier transform of the delta function $|k\rangle$, and is uniform in magnitude, so the distribution is determined only by $F\sum_s c_S|s\rangle$.

The other property used is that the Fourier transform of a state that is uniform over a normal subgroup of a finite group is uniform over representations that contain the subgroup in their kernel [HRTS00]. The subgroup can then be computed (efficiently when the group is abelian) from polynomially many samples.

In our algorithms we will use the first property and prove a new generalization of the second property.

2.2 Algebraic Number Theory Background

This section is intended to give a short overview of the background and the objects that appear. An introduction to the subject when d is assumed to be square-free can be found in [Joz03]. For the general non-square case, background can also be found in [BTW95, Len82b, Coh93].

The integer d is the input to our algorithm so the input size is $\log d$, and by "polynomial-time" we mean polynomial in $\log d$.

For a positive non-square integer Δ, $K = \mathbb{Q}(\sqrt{\Delta})$ is called a *real quadratic number field.* It is the set of numbers $\{u + v\sqrt{\Delta} : u, v \in \mathbb{Q}\}$ and it is a field. When Δ is also congruent to 0 or 1 modulo 4 it is called a *quadratic discriminant*, and the *order $\mathcal{O}$ of discriminant* Δ is the subring $\mathcal{O} = \mathbb{Z}[\frac{\Delta+\sqrt{\Delta}}{2}] \subseteq K$. It consists of the elements $\{a + b\frac{\Delta+\sqrt{\Delta}}{2} : a, b \in \mathbb{Z}\}$, and each $\alpha \in \mathcal{O}$ has a unique representation as $\alpha = \frac{a+b\sqrt{\Delta}}{2}$, where $a, b \in \mathbb{Z}$. The *norm* of an element $\frac{a+b\sqrt{\Delta}}{2}$ is defined by $\frac{a+b\sqrt{\Delta}}{2} \cdot \frac{a-b\sqrt{\Delta}}{2} = \frac{a^2-b^2\Delta}{4}$. The *units* of $\mathcal{O}$ are the invertible elements in $\mathcal{O}$. They have norm ± 1 and have the form $\pm\varepsilon^k$, where $k \in \mathbb{Z}$, and ε is called a *fundamental unit.* The fundamental unit can be chosen so that $\varepsilon > 1$ and with that choice the *regulator* of $\mathcal{O}$ is defined to be $R = \ln \varepsilon$. The fundamental unit in this representation can have exponentially many bits, so more compact representations, such as $\lfloor R \rceil$, the closest integer to the regulator of $\mathcal{O}$, are computed. The regulator satisfies $R \leq \sqrt{\Delta}\log\Delta$ [Len82a], so $\lfloor R \rceil$ requires only a number bits which is polynomial in $\log\Delta$ to represent. We will show how to compute $\lfloor R \rceil$, in quantum polynomial-time.

To solve Pell's equation, given an algorithm to compute $\lfloor R \rceil$ when given an order, we proceed as follows. Given an instance d of Pell's equation, compute $\lfloor R \rceil$ of the order with discriminant $\Delta = 4d$, which satisfies $\mathcal{O} = \mathbb{Z}[\sqrt{d}]$. Let x_0, y_0 be such that $\varepsilon = e^R = x_0 + y_0\sqrt{d}$ is the fundamental unit of $\mathcal{O}$. Then the norm of ε is $(x_0 + \sqrt{d}y_0)(x_0 - \sqrt{d}y_0) = x_0^2 - dy_0^2 = \pm 1$. To be a solution of Pell's equation we need to restrict to the norm one case. Even though the norm equation involves numbers potentially to large to compute with in polynomial-time it is possible to compute the norm in polynomial time from $\lfloor R \rceil$. This can be done by first computing R to a higher precision using classical algorithms (discussed below), and computing the compact representations and using the norm algorithm of [BTW95, Theorem 5.5]. If the computed norm is 1, then $\lfloor R \rceil$ is the solution to Pell's equation since $x_0^2 - dy_0^2 = 1$. If the norm is -1, then using the fact that norm is multiplicative, ε^2 generates the solutions of Pell's equation, so $2R$ is the solution which can be computed to high precision from $\lfloor R \rceil$ using classical algorithms.

Next we discuss how to gain access to information about the regulator R given an order $\mathcal{O}$. The *product* of two subsets $I, J \subseteq K$ is the additive subgroup of K generated by $\{xy : x \in I, y \in J\}$. An *invertible* $\mathcal{O}$-ideal I is a subset of K such that $\mathcal{O}I = I$ and for which there exists a subset $J \subseteq K$ with $IJ = \mathcal{O}$. The set of invertible ideals of $\mathcal{O}$ form an abelian group under multiplication and will be denoted $\mathcal{I}$. A *principal* ideal I is an additive subgroup of K of the form $\mathcal{O}\alpha$, with $\alpha \in K$. The set of principal ideals will be denoted $\mathcal{P}$, and it is a subgroup of $\mathcal{I}$. Finally, we need the set of reduced ideals $\mathcal{R}$. Let $x\mathbb{Z} + y\mathbb{Z}$ denote the set $\{xa + yb : a, b \in \mathbb{Z}\}$. Let $\tau(b, a)$ denote the unique integer τ such that $\tau \equiv b \mod 2a$, $-a < \tau \leq a$ if $a > \sqrt{\Delta}$, and $\sqrt{\Delta} - 2a < \tau < \sqrt{\Delta}$ if $a < \sqrt{\Delta}$. An invertible ideal I has the form

$$I = \frac{m}{l}\left(a\mathbb{Z} + \frac{b + \sqrt{\Delta}}{2}\mathbb{Z}\right) \quad (1),$$

where $l \in \mathbb{N}, a, b, m \in \mathbb{Z}, a, m > 0, b = \tau(b, a)$, and $4a$ divides $\Delta - b^2$ [BTW95]. An ideal I is *reduced* when 1 is a minimum in the ideal, which is a technical condition defined in [BTW95]. When an ideal is reduced, $m = 1$ and $a = l$. Furthermore, $a, |b| < \sqrt{\Delta}$. It follows that $\mathcal{R}$ is finite and has at most 2Δ elements. $\mathcal{R}$ is not a group, but is group-like under multiplication (as defined below).

The distance function $\delta : \mathcal{P} \to \mathbb{R}/R\mathbb{Z}$ is defined by $\delta((u + v\sqrt{\Delta})\mathcal{O}) = \frac{1}{2}\ln\left|\frac{u+v\sqrt{\Delta}}{u-v\sqrt{\Delta}}\right| \mod R$ [Len82b]. The *unit ideal* $\mathcal{O}$ has distance $\delta(1 \cdot \mathcal{O}) = \frac{1}{2}|\ln 1| = 0$. The function is well defined since for a unit ε, $\delta(\varepsilon\mathcal{O}) = \frac{1}{2}\left|\ln\frac{\varepsilon}{\varepsilon^{-1}}\right| = \frac{1}{2}\left|\ln\varepsilon^2\right| = 0 \mod R$. The *class group* is the finite abelian group $\mathrm{Cl} = \mathcal{I}/\mathcal{P}$. It provides a measure of how far $\mathcal{O}$ is from being a principal ideal domain (PID), in the sense that it $\mathcal{O}$ is a PID if and only if Cl is trivial.

The main two operations performed on $\mathcal{I}$ are composition $(\cdot)$ and reduction (ρ). There is also an operation which combines the two called multiplication $(*)$. It is not known how to compute the distance between two ideals in polynomial time if they are given in standard representation (1), (this would solve the principal ideal problem), but given an ideal I, it is possible to compute the distance to a new ideal J, without reduction modulo R which is unknown, if J is computed from I using composition and reduction. In particular, if the distance of I from $\mathcal{O}$ is known, and J is computed from I using composition and reduction, then the distance of J from $\mathcal{O}$ can be computed. These distances can be computed to a rough accuracy, say the closest integer, and later computed to a greater precision. To achieve this it is only necessary to keep track of the sequence of composition and reduction steps, and to run the same sequence of composition and reduction steps again using the higher precision.

The *composition* of two ideals $I, J \in \mathcal{I}$ is their product $I \cdot J \in \mathcal{I}$. The distance function satisfies $\delta(I \cdot J) = \delta(I) + \delta(J)$. *Reduction* is a map from $\mathcal{I}$ to $\mathcal{I}$, and after a polynomial number of steps, the ideal will be in $\mathcal{R}$ (reduced). We will not give the formula for the reduction step here, but $\delta(I) + \frac{1}{\sqrt{\Delta}} \leq \delta(\rho(I)) \leq \delta(I) + \frac{1}{2}\ln\Delta$. Also, two applications has a constant minimum distance: $\delta(\rho^2(I)) > \delta(I) + \ln 2$. The effect of reduction on I is to multiply it by an element in K, so reduction preserves the ideal class, i.e. the member of the class group. *Multiplication* $* : \mathcal{R} \to \mathcal{R}$ takes reduced ideals I, J, and $I * J$ is computed by first applying composition, and then applying reduction a polynomial number times k until it is reduced. Therefore $\delta(I * J) = \delta(\rho^k(I \cdot J)) = \delta(I \cdot J) + (\delta(\rho^k(I \cdot J)) - \delta(I \cdot J)) = \delta(I) + \delta(J) + (\delta(\rho^k(I \cdot J)) - \delta(I \cdot J))$. The last term is at most a polynomial and it is in this sense that $\mathcal{R}$ is group-like. If the last term was zero, then distances would add, and δ would be a homomorphism on $\mathcal{R}$. However, it is only true that δ is a homomorphism on $\mathcal{I}$. Also, multiplication of two elements in $\mathcal{R}$ is commutative but not associative.

The reduction step ρ restricted to $\mathcal{R}$ is a permutation, and the equivalence class modulo $\mathcal{P}$ is preserved (e.g. a principal ideal stays principal) after application. In fact, it is possible to cycle through the set of all reduced principal ideals, called the *principal cycle*, by picking one and applying powers of ρ to it. This is also true for any set of reduced ideals modulo $\mathcal{P}$.

Algorithm 2.1 (Computing a reduced ideal to the left of x)**.**
Input: A rational number x.
Output: The reduced ideal I to the left of or at $x \bmod R$ and its distance to x.

1. *Apply ρ twice to $\mathcal{O}$ to compute an ideal I' and its distance $\delta(I')$. The distance $\delta(I')$ is at least a constant and at most a polynomial in $\log \Delta$.*

2. *Use repeated squaring of the operation $*$ to compute an ideal I'' within a polynomial distance of x and its distance δ'' to x.*

3. *Use ρ to search the region for the appropriate ideal I and its distance δ to x.*

The distance is kept at some chosen precision. Since any pair of ideals is separated by at least $1/\sqrt{\Delta}$, a polynomial number of precision bits suffices for distinguishing different ideals. For any ideal J, principal or not, and distance $x \in \mathbb{Q}$, the ideal J' that is x away from J can also be computed in a similar manner: first compute the ideal I at distance x from $\mathcal{O}$, and then multiply J and I.

When computing the ideal to the left of or at x, a technical problem arises when the ideal has a distance too close to x and the precision used cannot distinguish whether the ideal is to the left or right of x. The third part of the next definition handles this problem.

The main functions we are interested in are the following:

Definition 2.1. *Let I_x be the reduced the ideal to the left of or at x. More generally, given a reduced ideal J and a distance x, let $I_{J,x}$ be the ideal at the largest distance less than or equal to x from J.*

1. *Define $f : \mathbb{R} \to \mathcal{I} \times \mathbb{R}$ by $f(x) = (I_x, \delta_x)$, where $\delta_x = x - \delta(I_x)$.*

2. *Define $f_J : \mathbb{R} \to \mathcal{I} \times \mathbb{R}$ by $f_J(x) = (I_{J,x}, \delta_{J,x})$, where $\delta_{J,x} = x - \delta(I^{-1}J)$.*

3. *Let $j, L \in \mathbb{Z}$. Define $f_{J,j} : \mathbb{R} \to \mathcal{I} \times \mathbb{R}$ by $f_{J,j}(x) = (I_{J,x+j/L}, \delta_{J,x+j/L})$.*

The first type of function is used, for example, in [BW89b]. The first function is a special case of the second one since $f = f_{\mathcal{O}}$. The second function is a special case of the third one since $f_J = f_{J,0}$. The function f is one-to-one in $[0, R)$, and has period R, the regulator. It is well-defined, and therefore one-to-one in $[0, R)$, because the distance between any two reduced ideals is positive. It has period R because the distance function is defined modulo R. For example, applying the reduction step to the ideal with greatest distance modulo R results in the unit ideal, and the function repeats. The function $f_{J,j}$ has the same properties, it is simply takes f and shifts it by the ideal J and by the distance j/L.

We will need to work with discretized versions of f and this is done by restricting the domain to the integers or rational numbers, and by computing the distance δ to some desired precision. One technical problem which arises is that conceivably an ideal has a distance so close to x that the precision used in the algorithm cannot determine if the ideal is on the left or right of x. In general there is no known upper bound on the precision to prevent this from happening. The result can be that an algorithm which tries to compute f on two different inputs incorrectly returns the same value. For example, consider an algorithm which attempts to compute $f(i/N)$ and $f((i+1)/N)$. If different repeated squaring steps are used and the rounding is different as a result, then the algorithm could return $(I, 0)$ for both, even though the function values differ.

The precision problem can be fixed using $f_{J,j}$. If $f_{J,j}$ is evaluated on integer multiples of $1/N$ and no ideal is close to an evaluation point i/N then there are no problems.

Claim 2.1. *Let $N, q \in \mathbb{Z}$, $L = qN16$. With probability $1 - 1/N$ over choices of $j \in \{0, \dots, L/N - 1\}$, if $i \in \{0, \dots, q-1\}$ then no reduced ideal is within $1/L$ of $i/N + j/L$.*

Proof. Limiting the domain to $[0, q/N]$ upper bounds the number of reduced ideals encountered to $q/N \cdot 2/\ln 2$ (including repeats). Because there are at at most q rational numbers i/N and at most $q/N \cdot 2/\ln 2$ reduced ideals, the probability that no ideal has a distance closer than $1/L$ to one of the rational numbers i/N is at least $1 - (q/N \cdot 2/\ln 2)/((L/2)/N) \geq 1 - 1/N$. □

It is possible to check if a given number x is within a polynomial distance of a multiple of the regulator. Given x, compute the ideal closest to it. Then apply ρ and ρ^{-1} a polynomial number of times and search for the unit ideal.

To summarize:

Theorem 1. *Let $\mathcal{O}$ be an order of discriminant Δ. Given Δ, let N, a, b be integers which are polynomial in Δ.*

1. *Let $q, N \in \mathbb{Z}$, $L = qN16$, J be a reduced ideal, and $f_{J,j}$ be as in Definition 2.1. Then with probability $1 - 1/N$ over choices of $j \in \{0, \dots, L/N\}$, $f_{J,j}(i/N)$ can be evaluated in time polynomial in $\log \Delta$ when $i \in \{0, \dots, q-1\}$.*

2. *For a distance $u = a/b \in \mathbb{Q}$ and a reduced ideal I, it is possible to check if $u \bmod R$ is within a polynomial of the distance of I in time polynomial in $\log \Delta$. In particular, it is possible to efficiently check if a rational number is near a multiple of the regulator, by using the unit ideal $\mathcal{O}$ as I.*

3 Computing the Regulator

In this section we develop an algorithm for computing the regulator. This will be done by finding the period of a periodic function defined on the reals. In Section 3.1, we show how to find the period of a function which is discrete but still has an irrational period. In Section 3.2 we show how to construct such a function for the regulator.

3.1 Approximating Irrational Periods of Functions

Given a function with an irrational period, we must first discretize it to use it in an algorithm. We start by defining the notion of pseudo-periodic to handle this.

Definition 3.1. *A function $f : \mathbb{Z} \to X$, where X is any set, is called* pseudo-periodic with period S, at offset k *if for each i either $f(k) = f(k + \lfloor iS \rfloor)$ or $f(k) = f(k + \lceil iS \rceil)$, where $S \in \mathbb{R}$. We write this condition as $f(k) = f(k + [iS])$ to denote either floor or ceiling. A function is* ε-pseudo-periodic with period S *if for at least an ε-fraction of offsets $k \in \{0, \ldots, \lfloor S \rfloor\}$, f is pseudo-periodic at offset k.*

Given a periodic function f with period $S \in \mathbb{Z}$ which is also injective in $\{0, \ldots, S-1\}$ it is easy to verify that an integer T is a multiple of the period by checking whether $f(0) = f(T)$. Given a function which is pseudo-periodic and injective on a subset of offsets for which it is pseudo-periodic, it is not quite as straightforward to verify the period. For the sake of simplicity we will assume there is an efficient way to check the period in the period finding algorithm, as will exist in our application in Section 3.2.

Definition 3.2. *A verification procedure for the period of a function with irrational period as in Definition 3.1 returns yes when the input T is within one of S, or alternatively, when T is within one of an integer multiple of S.*

The idea of the algorithm for pseudo-periodic functions is the same as the original period finding algorithm: the Fourier transform inverts the period. In this case the superposition created during Fourier sampling is no longer periodic but only pseudo-periodic. Following the general idea however, if we Fourier sample f over $\mathbb{Z}_q$ and observe an integer c, then c should be close to an integer multiple of the irrational number q/S, and from this we will recover a value close to S. If computing with irrational numbers were possible, then given two random integer multiples $k\alpha$ and $l\alpha$, the irrational number α can be computed by dividing the two numbers to get k/l, and then dividing $k\alpha$ by k. This idea works if k and l are relatively prime, which happens with inverse polynomial probability. Fourier sampling f actually results in rounded versions of $k\alpha$ and $l\alpha$, but we will show that k/l will appear in the continued fraction expansion of the ratio of the two measured values.

Algorithm 3.1 (Approximate the period of a pseudo-periodic function).
Input: An ε-pseudo-periodic function f with period $S \in \mathbb{R}$ such that if k_1 and k_2 are among the ε-fraction of pseudo-periodic offsets then $f(k_1) \neq f(k_2)$, an efficient verification procedure as in Definition 3.2, and upper bound M on the period S.
Output: An integer within one of S.

```
1. Choose an integer $q \geq 3M^2$.
2. Fourier sample $f$ over $\mathbb{Z}_q$ twice resulting in values $c$ and $d$.
3. Compute the continued fraction expansion of $c/d$.
4. For each convergent $c_i/d_i$, test whether $\lfloor c_i q/c \rceil$ is a multiple of the period.
```

5. *Output the smallest value $\lfloor c_i q/c \rceil$ that was a multiple of the period in Step 4.*

In the algorithm an upper bound on S is needed, however the standard method of circumventing this can be used. Assume S is at most two and run the algorithm, if the answer is wrong, double the bound on S and repeat the algorithm. In a polynomial in $\log S$ number of steps a correct bound will be used and the algorithm will return the correct answer.

Lemma 3.1. *Given an ε-pseudo-periodic function with period $S \in \mathbb{R}$ greater than some absolute constant such that if k_1 and k_2 are among the ε-fraction of pseudo-periodic offsets then $f(k_1) \neq f(k_2)$, an efficient verification procedure for the period, and an upper bound M on S, Algorithm 3.1 computes an integer a such that $|S-a| \leq 1$ in time polynomial in $\log S$, with probability $\Omega(\varepsilon^2/(\log M)^4)$.*

Proof. One Fourier sampling step starts by evaluating the function in superposition and measuring the function value. Suppose this results in measuring $f(m)$, where $m \leq S$. Since f is pseudo-periodic at an ε fraction of offsets, with probability at least ε, f is pseudo-periodic at m. If f is not pseudo-periodic at m then the rest of the algorithm will be completed and some integer will be returned, which will probably fail the verification procedure.

Let $p = \lfloor (q-m)/S \rfloor$. If f is pseudo-periodic at offset m then the resulting superposition is $\frac{1}{\sqrt{p}} \sum_{i=0}^{p-1} |m + [iS]\rangle$. Since we are Fourier sampling, we may assume w.l.o.g. that $m = 0$. Therefore the measured distribution will be the same as the one induced by Fourier sampling $\frac{1}{\sqrt{p}} \sum_{i=0}^{p-1} |[iS]\rangle$.

The Fourier transform at $|j\rangle$ has amplitude $\frac{1}{\sqrt{pq}} \sum_{i=0}^{p-1} \omega_q^{j[iS]}$. This is similar to a geometric series except that each term iS rounded to $[iS]$. Let $[iS] = iS + \delta_i$, where $-1 < \delta_i < 1$. Let $j = k\frac{q}{S} + \varepsilon$, for $k \in \{0, \ldots, \lfloor S \rfloor\}$, and $-\frac{1}{2} \leq \varepsilon \leq \frac{1}{2}$. Then the fraction in the $\omega_q^{j[iS]}$ term is

$$\frac{j[iS]}{q} = \left(\frac{k}{S} + \frac{\varepsilon}{q}\right)(iS + \delta_i) = \frac{\varepsilon i S}{q} + \frac{k\delta_i}{S} + \frac{\varepsilon \delta_i}{q} \pmod 1.$$

When $j \leq q/2\log M$, $|k\delta_i/S| \leq 1/2\log M$ and the rounding term in $[iS]$ is $|k\delta_i/S| + |\varepsilon\delta_i/q| \leq 2/\log M$. The condition in Claim 3.1 states that $i\delta/M$ should be at most $3/4$, and in the present case $i \leq q/S$, so we have $|\varepsilon i S/q| \leq 1/2$. For p we have $p = \lfloor \frac{q-m}{S} \rfloor \geq q/S - 2 \geq 3M^2/S - 2 \geq 3M - 2$. When $M \geq 2^{262}$ we can apply Claim 3.1,

$$\left| \frac{1}{\sqrt{pq}} \sum_{i=0}^{p-1} \omega_q^{j[iS]} \right|^2 \geq \frac{1}{pq} cp^2,$$

where c is a constant. Then the probability of measuring an integer of the form $\lfloor k\frac{q}{S} \rceil$ is at least $cp/q \geq c/(2S)$.

There are $S/\log M$ integer multiples of q/S less than $q/\log M$ so the probability of measuring two values less than $q/\log M$ is at least $\Omega(1/(\log M)^2)$. Furthermore, the probability that the two values are relatively prime is at least $\Omega(1/(\log(S/\log M))^2)$ by the prime number theorem. The probability of measuring two such values satisfying all the conditions is therefore at least $\Omega(\epsilon^2/(\log M)^4)$.

Given that $c = \lfloor kq/S \rceil$ and $d = \lfloor lq/S \rceil$, we now show that k/l is a convergent in the continued fraction expansion of c/d. We will use the fact that if x is any irrational number, $a/b \in \mathbb{Q}$, and $|x - a/b| \leq \frac{1}{2b^2}$, then a/b is a convergent in the continued fraction expansion of x [Sch86]. We will show that $|\frac{c}{d} - \frac{k}{l}| \leq \frac{1}{2l^2}$. Given this, choose an irrational number x between c/d and k/l that is within $\frac{1}{2d^2}$ of c/d. Then k/l and c/d are convergents of the continued fraction expansion of x, and therefore k/l is a convergent of the continued fraction expansion of c/d.

Let $c = kq/S + \varepsilon_k$ and $d = lq/S + \varepsilon_l$, with $-1/2 \leq \varepsilon_k, \varepsilon_l \leq 1/2$, and $k \leq l \leq S$. Then

$$\left|\frac{c}{d} - \frac{k}{l}\right| = \left|\frac{kq + \varepsilon_k S}{lq + \varepsilon_l S} - \frac{k}{l}\right| = \left|\frac{S(\varepsilon_k l - \varepsilon_l k)}{l^2 q + \varepsilon_l S l}\right|$$

$$\leq \left|\frac{S(l+k)}{2l^2 q - 2Sl/2}\right| \leq \frac{S}{lq - S/2},$$

which is at most $\frac{1}{2l^2}$ if $q \geq 3S^2$. The second to last inequality uses the worst case, which is $\varepsilon_k = 1/2$ and $\varepsilon_l = -1/2$.

Finally, if $c = \lfloor kq/S \rceil$ and $q \geq S^2$ then $|S - \lfloor kq/c \rceil| \leq 1$. □

We now prove the claim used in the lemma, which is a bound about sums that are close to geometric series, where close is in the sense that it is a geometric series if the function f below satisfies $f = 0$.

Claim 3.1. *Let $M \in \mathbb{Z}$ satisfy $\log M \geq 262$, $\delta \in [-3/4, 3/4]$ be a constant, f a function such that $-1/\log M \leq f(i) \leq 1/\log M$ for all $i \in \{0, \dots, M-1\}$. There exists a constant c such that*

$$\left|\sum_{i=0}^{M-1} \omega^{\delta i/M + f(i)}\right|^2 \geq cM^2,$$

where $\omega^{a/b}$ is defined to be ω_b^a for $a, b \in \mathbb{Z}$.

Proof. The angle $\delta i/M + f(i)$ covers at most $3/4 + 2/\log M$ of the complex circle. Orient the vectors so that the length of the sum of the vectors below the real axis is minimized. This can be done so that the $1/4 - 2/\log M$ fraction of the circle with no vectors will be below the axis. Consider only the complex part (y-component) of each vector. There are at most $(1/4 + 2/\log M)M/\delta$ vectors with a negative component. A $1/8$ fraction have y-component at most $1/\sqrt{2}$, while the rest have y-component at most 1. The total y-component below the axis is then at most $M/\delta(\frac{1}{8}\frac{1}{\sqrt{2}} + (\frac{1}{8} + \frac{2}{\log M}))$.

In the upper half-plane, there are at least $1/6 - 2/\log M$ fraction of vectors with y-component at least $\sqrt{3}/2$, and $1/6$ with y-component at least $1/2$. The y-component of the sum of these vectors is at least $M/\delta(\frac{\sqrt{3}}{2}(\frac{1}{6} - \frac{2}{n}) + \frac{1}{2}\frac{1}{6})$. For M such that $\log M \geq 262$, the sum of these are larger than those in the lower half-plane.

There are at least a constant fraction, say $1/12$ of the vectors left with at least a constant y-coordinate. □

3.2 Application to the Regulator

Suppose a positive non-square integer Δ congruent to 0 or 1 mod 4 is given and we want to compute the regulator R of the order $\mathcal{O}$ of discriminant Δ. By Theorem 1 we have access to a function which is pseudo-periodic. Recall Definition 2.1 and suppose for a given integer i, $f(i/N) = (I, \delta)$. Define $f_N : \mathbb{Z} \to \mathcal{I} \times \mathbb{Z}$ by $f_N(i) = (I, k)$, where $k = \lfloor N\delta \rfloor$.

Lemma 3.2. *If $N \geq 2\sqrt{\Delta}$ then f_N is 1/2-pseudo-periodic with period $NR \in \mathbb{R}$. The function f_N also has the uniqueness property on the function on different cosets required by Lemma 3.1.*

Proof. The minimum distance between ideals is $1/\sqrt{\Delta}$, so by choice of N, each maximal set of consecutive integers that map to a given ideal has size at least 2. Now consider a fixed ideal I and suppose the distance to the next ideal is δ. Let i be such that $f_N(i) = (I, 0)$. Then for $k \in \{0, \ldots, \lfloor \delta N \rfloor - 1\}$, $f_N(i+k) = (I, k)$, because $(i+k)/N \leq (i + \lfloor \delta N \rfloor - 1)/N \leq (i-1)/N + \delta$, which is less than the distance of the next ideal. In the case where $(i + \lfloor \delta N \rfloor)/N$ is less than the distance to the next ideal, then $f_N(i + \lfloor \delta N \rfloor) = (I, \lfloor \delta N \rfloor)$. When δN is not an integer then f_N cannot return I on $\lceil \delta N \rceil$ since $(i + \lceil \delta N \rceil)/N > i/N + \delta$, which is the distance to the next ideal. In summary, a maximal set of consecutive integers mapping to ideal I, with distance δ to the next ideal, either maps to the set $\{(I, 0), (I, 1), \ldots, (I, \lfloor \delta N \rfloor - 1)\}$ or to the set $\{(I, 0), (I, 1), \ldots, (I, \lfloor \delta N \rfloor)\}$.

Now suppose k is an integer less than NR such that f_N maps k to ideal I which has distance δ to the next ideal and $f_N(k) \neq (I, \lfloor \delta N \rfloor)$. For each maximal set of consecutive integers mapping to I, this happens for at least a $1 - 1/2 = 1/2$ fraction of all integers in the set mapping to I (all except possibly the last one), so the same fraction holds across all k at most NR.

Let k be such an integer. It must be verified that $f_N(k) = f_N(k + [iNR])$ for all i. If the ideal to the left of the rational number k/N is I, then the same is true of the real number $(k + iRN)/N = k/N + iR$. If the distance from I to k/N mod R is greater than the distance from I to $k/N + iR$ mod R then $f_N(k) = f_N(k + \lfloor iRN \rfloor)$. If the distance from I to k/N mod R is less than the distance from I to $k/N + iR$ mod R, then $f_N(k) = f_N(k + \lceil iRN \rceil)$.

The function clearly satisfies the uniqueness condition. □

This reduces approximating the regulator to approximating the period of the function f_N, whose period is $NR \in \mathbb{R}$. Theorem 1 states that $f_N(i)$ can be computed efficiently provided that no ideal has distance too close to i/N.

Let $f_{N,j} = f_{\mathcal{O},j}$ from Definition 2.1.

Claim 3.2. *If $N \geq 2\sqrt{\Delta}$ then the functions $\{f_{N,j}\}_j$ are 1/2-pseudo-periodic with period $NR \in \mathbb{R}$.*

Proof. Shifting the function by j/L effectively changes the distance of all the ideals by j/L. This does not change any properties of the function, so use the fact that $f_N = f_{N,0}$ and apply Lemma 3.2. □

Theorem 2. *There is a polynomial-time quantum algorithm that, given a quadratic discriminant Δ, approximates the regulator to within δ of the associated order $\mathcal{O}$ in time polynomial in $\log \Delta$ and $\log \delta$ with probability exponentially close to one.*

Proof. By Claim 3.2, with high probability over choices of j, $f_{N,j}$ satisfies the conditions of Lemma 3.1. Using Algorithm 3.1 compute an integer within 1 of NR. Polynomial in $\log \Delta$ repetitions will boost the probability of correctness to exponentially close to 1. This approximation on R can be improved by using classical algorithms. □

4 The Principal Ideal Problem

In this section we show how to compute the distance of a principal ideal, or more generally, how to compute the generator of one ideal relative to another. This solves the principal ideal problem in real quadratic number fields, which is to decide if an ideal $I \subseteq \mathcal{O} \subseteq K$ is principal, and if it is to find the generator α such that $\alpha\mathcal{O} = I$. As with the fundamental unit, the generator α may be too large to write down in polynomial-time, so instead an approximation of the distance $\frac{1}{2}\log\frac{\alpha}{\bar{\alpha}} \bmod R$ of I is computed. To test if an ideal is principal, run the distance finding algorithm to compute a candidate distance and check if the ideal really has that distance using Theorem 1. If the ideal is not reduced, then it should be reduced first.

One application of this quantum algorithm is breaking the Diffie-Hellman protocol based on real quadratic number fields proposed in [BW89b]. The protocol assumes that given $f_N(i)$ for a random integer $i \in \{0, \ldots, \lfloor NR \rfloor\}$, it is not possible to efficiently compute i. Since we can compute the distance of the ideal, we can invert this function.

The general idea for the algorithm is the same as in Shor's discrete log algorithm, but we have new technical difficulties because we are computing modulo an irrational number. It is also different because there is not an underlying group, only a group-like set. In the original algorithm, given a prime p, a generator $g \in \mathbb{Z}_p$, and an element $g^r \in \mathbb{Z}_p$, the function Fourier sampled is $g(a,b) = g^{ar-b}$. This problem can be viewed as a hidden subgroup problem in $\mathbb{Z}_{p-1} \times \mathbb{Z}_{p-1}$, where the subgroup is $\{(a, ar) \in \mathbb{Z}_{p-1} \times \mathbb{Z}_{p-1}\}$. Only the subgroup needs to be analyzed, not a general coset, because we are Fourier sampling.

Recall the definition of $f : \mathbb{R} \to \mathcal{I} \times \mathbb{R}$ from Section 2.2. The function f is periodic with period R, and $f(x)$ is the reduced ideal to the left of x together with the distance from x to the ideal.

To set up the principal ideal problem, given a reduced ideal I_x at distance x and with N chosen as in Lemma 3.1, define the function $h : \mathbb{Z} \times \mathbb{Z} \to \mathcal{I} \times \mathbb{Z}$ by $h(a,b) = (I_{ax+b/N}, \lfloor N\delta_{ax+b/N} \rfloor)$, where x plays the role of r and is a real number instead of an integer. The function h can be efficiently computed as follows. Given the ideal I_x, first compute I_x^a by repeated squaring using composition and reduction and keeping track of errors. The result will be the ideal the to the left of $ax \bmod R$ together with distance δ_1 to $ax \bmod R$. Next compute the ideal to the left of b/N, together with the distance δ_2 to b/N. Next multiply these two ideals and use δ_1 and δ_2 and the reduction operator to get the ideal to the left of $ax + b/N \bmod R$ together with the error $\delta_3 = \delta_{ax+b/N}$ (see [BW89b]). Notice this only works when a is an integer, since the given ideal I_x can only be raised to an integer power. The set that is analogous to the subgroup in the finite case is the set $\{(a,b) \in \mathbb{Z} \times \mathbb{Z} : ax + b/N (\bmod R) < 1/N\}$, where R is the regulator. We choose the integer N, and compute the subgroup element $(1, b/N)$, where b/N will approximate x.

Computing I_{ax} does not have error problems, but computing $I_{b/N}$ may have the same problems mentioned in the background section. These are similarly fixed by choosing a random integer j for the algorithm and, then on on arbitrary input a, b, evaluating $f_{I_{ax},j}(b/N)$.

Given an ideal I we can determine if it is principal, and if it is compute its distance x. This is done by assuming it is principal, computing its distance, and then verifying the result. If the distance of the ideal is not close to the computed distance then it is not principal.

Algorithm 4.1 (Compute the distance of an ideal)**.**
Input: A quadratic discriminant Δ, and an ideal I in the associated order $\mathcal{O}$.
Output: An approximation of the distance of I if I is principal, and "No" if not.

1. *Approximate the regulator R using Theorem 2 to the desired precision.*

2. *Let $M = \lceil 2R \rceil + 1$. Compute an integer $N > 2\sqrt{\Delta}$ such that $|M\lfloor RN \rceil - MRN| \leq 1/4$, where $\lfloor \cdot \rceil$ is the closest integer function as follows. Let $B = \lceil 2\sqrt{\Delta} \rceil$, and compute the continued fraction expansion of BR to find a rational number p/q such that $|BR - p/q| \leq 1/(q4M)$. Let $N = qB$.*

3. *Fourier sample h twice over $\mathbb{Z}_{M\lfloor NR \rceil} \times \mathbb{Z}_{\lfloor NR \rceil}$ to get samples (c_1, d_1) and (c_2, d_2).*

4. *Compute integers a and b such that $ad_1 + bd_2 = 1$.*

5. *Compute $(ac_1 + bc_2)/(NM)$, and then reduce modulo R. The result will be within 1 of x, which can be verified using Theorem 1. Output "No" if the computed distance is not the distance of I.*

Theorem 3. *The above algorithm approximates the distance of a principal ideal in time polynomial in $\log \Delta$ when the regulator is larger than some absolute constant. The algorithm is successful with probability $\Omega(1/\log(\Delta))$. Polynomial in $\log \Delta$ repetitions gives probability exponentially close to one.*

Proof. By Direchlet's theorem the choice of N in Step 2 exists where $q \leq 4M$. The numbers p and q can be computed in polynomial time using the continued fraction algorithm.

Let x be the distance of the given reduced ideal I_x. The set that is analogous to the subgroup in this case is the set

$$\{(a, b) \in \mathbb{Z} \times \mathbb{Z} : ax + b/N - \gamma_a/N \equiv 0 \pmod{R}, 0 \leq \gamma_a < 1\}.$$

Since we are Fourier sampling, assume w.l.o.g. that the superposition is over this set, i.e.,

$$\frac{1}{\sqrt{M\lfloor RN \rceil}} \sum_{a=0}^{M\lfloor RN \rceil - 1} \left| a, \left\lceil \frac{ax}{R} \right\rceil RN - axN + \gamma_a \right\rangle .$$

This happens when the last integer between two ideals is not measured, which happens with probability 1/2.

We will show that with large probability, a sample (c,d) is such that $\frac{c}{NM} - \frac{\gamma_d}{NM} \equiv dx \bmod R$, where $-1/2 \leq \gamma_d \leq 1/2$. Rewriting this condition, let $c = dxNM - \lfloor \frac{dx}{R} \rfloor RNM + \gamma_d$. The Fourier transform at $|c,d\rangle$ is

$$\frac{1}{M\lfloor RN\rceil\sqrt{\lfloor RN\rceil}} \sum_{a=0}^{M\lfloor RN\rceil -1} \omega_{M\lfloor RN\rceil}^{ac+bdM}.$$

Writing out $ac + bdM$ we get

$$adxNM - a\left\lfloor \frac{dx}{R} \right\rfloor RNM + a\gamma_d + d\left\lceil \frac{ax}{R} \right\rceil RNM - adxNM + dM\gamma_a$$
$$= \left(d\left\lceil \frac{ax}{R} \right\rceil - a\left\lfloor \frac{dx}{R} \right\rfloor\right) RNM + a\gamma_d + dM\gamma_a.$$

Let $\lambda = \lfloor RN\rceil$. Reducing modulo $M\lfloor RN\rceil = M(RN+\lambda)$ gives $-\lambda M(d\lceil \frac{ax}{R}\rceil - a\lfloor\frac{dx}{R}\rfloor) + a\gamma_d + dM\gamma_a$. Rewriting $\lceil \frac{ax}{R} \rceil$ as $\frac{ax}{R} + \delta_a$ and $\lfloor \frac{dx}{R} \rfloor$ as $\frac{dx}{R} - \delta_d$, with $0 \leq \delta_a, \delta_d < 1$, we get $a(\gamma_d + \lambda M\delta_d) + dM(\gamma_a - \lambda\delta_a)$.

By choice of N and M, $\lambda \leq 1/(4M)$, so $a|\gamma_d + \lambda M\delta_d|/(M\lfloor RN\rceil) \leq |\gamma_d + \lambda M\delta_d| \leq 3/4$. When $d \leq \lfloor RN\rceil / \log(M\lfloor RN\rceil)$ it follows that $dM(\gamma_a - \lambda\delta_a)/(M\lfloor RN\rceil) \leq 1/\log(M\lfloor RN\rceil)$. By Claim 3.1 the probability of seeing such a sample $d \leq \lfloor RN\rceil/\log(M\lfloor RN\rceil)$ is at least $\Omega(1/\lfloor RN\rceil)$.

There are $\lfloor RN\rceil$ values d so if $d \leq \lfloor RN\rceil/\log(M\lfloor RN\rceil)$ then d is measured with probability $\Omega(1/\log(M\lfloor RN\rceil))$. The probability of measuring two relatively prime values using this procedure is at least $\Omega(\varepsilon^2/\log(M\lfloor RN\rceil)^4)$, where $\varepsilon = 1/2$. Since MRN is polynomial in Δ, this is with probability at least $\Omega(1/(\log\Delta)^4)$.

For the classical reconstruction, suppose we start with such a pair. Then

$$c_1 = d_1xNM - \left\lfloor \frac{d_1x}{R} \right\rfloor RNM + \gamma_{d_1},$$

$$c_2 = d_2xNM - \left\lfloor \frac{d_2x}{R} \right\rfloor RNM + \gamma_{d_2},$$

and

$$ac_1 + bc_2 = xNM - \left(a\left\lfloor \frac{d_1x}{R} \right\rfloor + b\left\lfloor \frac{d_2x}{R} \right\rfloor\right) RNM + a\gamma_{d_1} + b\gamma_{d_2}.$$

After dividing by NM we have,

$$x - \left(a\left\lfloor \frac{d_1x}{R} \right\rfloor + b\left\lfloor \frac{d_2x}{R} \right\rfloor\right) R + (a\gamma_{d_1} + b\gamma_{d_2})/(NM).$$

Since a and b are at most the maximum of d_1 and d_2,

$$(a\gamma_{d_1} + b\gamma_{d_2})/(NM) \leq 2\lfloor RN\rceil/(NM) \leq 1$$

by our choice of M. Therefore reducing modulo R, where R has high enough precision, will result in $x \pm 1$, which can be verified and recomputed to a better accuracy using classical algorithms. □

5 Class Group Computations

In this section we show how to compute the class group G and class number of a real quadratic number field. The class group G is a finite abelian group and given a set of generators, the task is to decompose it as $G = \times\mathbb{Z}_{e_i}$ such that $e_i|e_{i+1}$. In its simplest form, when there is a unique bit string representing each element in the group, decomposing a finite abelian group reduces to a hidden subgroup problem. The problem we must deal with here is that it is not known how to efficiently compute a unique representative for an element in the class group. After reviewing how the unique representation case reduces to the hidden subgroup problem, we will show how to use then algorithm from Section 4 to create a superposition over an arbitrary equivalence class, allowing the HSP algorithm to carry through.

Decomposing a finite abelian group given a set of generators $g_1, \ldots, g_k$ reduces to a hidden subgroup problem over $\mathbb{Z}^k$ as follows. The HSP instance is the function $f(e_1, \ldots, e_k) = \sum_{i=1}^{k} e_i g_i$. The hidden subgroup is the lattice of relations $L \subseteq \mathbb{Z}^k$ where $[e_1, \ldots, e_k] \in L$ iff $\sum_i e_i g_i = 0$. The hidden subgroup problem algorithm can be used to find a basis matrix B whose columns span L. The Smith normal form of B then reveals the group structure. In particular, in (classical) polynomial-time, unimodular matrices U, V are computed such that $UBV = D$, where D is a diagonal matrix. The elements on the diagonal of D are $[e_1, e_2, \ldots, e_k, 1, \ldots, 1]$ where $e_i|e_{i+1}$, which are the orders of the cyclic factors of $G \cong \times_i \mathbb{Z}_{e_i}$, and $[g'_1, \ldots, g'_n] = [g_1, \ldots, g_n]U^{-1}$ gives a basis for the group.

Recall the definitions of $\mathcal{I}$ and $\mathcal{P}$ from Section 2.2. $\mathcal{I}$ is the set of invertible ideals and is a group under multiplication $(\cdot)$. $\mathcal{P}$ is the set of principal invertible ideals, and is a subgroup of $\mathcal{I}$. The *class group* is defined as the quotient group $\mathrm{Cl} = \mathcal{I}/\mathcal{P}$. The set of reduced ideals $\mathcal{R}$ is a finite subset of $\mathcal{I}$. Any ideal $I \in \mathcal{I}$ and the ideal resulting after reduction are in the same equivalence class modulo $\mathcal{P}$. Therefore, the class group is finite since $\mathcal{R}$ is. Just as there is a cycle of reduced principal ideals, there is a cycle of reduced ideals for each group element. This cycle can have exponential size, and is different for each element in the class group in that it can have a different number of reduced ideals and a different set of distances between all the ideals in the class. In addition, only a relative distance is defined between two ideals (given two equivalent ideals I and J, defined as the distance of $I^{-1}J$), so in general there is no ideal to single out with distance zero.

It is not known how to test identity efficiently in the class group classically. An efficient quantum identity test does follow from our principal ideal test, since given two ideals I and J, it can be tested if $I^{-1}J$ is principal. Assuming the GRH, it is possible to compute a polynomial size set of generators for the class group in polynomial time [Coh93]. The best classical algorithms for computing the regulator compute the class group at the same time [Buc89]. The standard quantum algorithm for decomposing a group cannot be directly applied since we do not have unique representatives for each group element. After multiplying two elements, we are left with a reduced ideal, but it could be any reduced ideal in the cycle.

Here we deal with the fact that we cannot define an HSP instance as above since each group element has a potentially exponential number of representatives (i.e. reduced ideals). The natural approach is to replace a basis state $|g\rangle$ with a superposition of elements in the equivalence class, preserving the property that two different class group elements are mapped to orthogonal vectors. Such an approach has been used before, such as in [Wat01]. Here we have the added difficulty that we are computing on the set of reduced ideals, which is not a group, and that each equivalence class has a different number of such ideals.

To use the standard HSP algorithm to decompose the group we will now show how to compute a quantum state $|\phi_{e_1,\ldots,e_k}\rangle$ given a set of coefficients $e_1, \ldots, e_k$ for the group element $\sum_i e_i g_i$. These states must satisfy two properties for the HSP algorithm to work. First, the states must have exponentially small inner product when the group elements are different. Second, if the same group element is computed from two different coefficient vectors, then the resulting states must have inner product exponentially close to one. Roughly speaking, this will be achieved by creating a superposition of all reduced ideals in the class. Ideals in different classes are different, so orthogonality is achieved. Some more care must be taken when ideals are in the same class and states will only be exponentially close.

Recall Definition 2.1 and suppose for a given reduced ideal I, $f_I(i/N) = (J, \delta)$. Let $N \in \mathbb{Z}$. Then define $f_{I,N} : \mathbb{Z} \to \mathcal{I} \times \mathbb{Z}$ by $f_{I,N}(i) = (J, k)$, where $k = \lfloor N\delta \rfloor$. Define the state $|\phi_{e_1,\ldots,e_k}\rangle = |\phi_I\rangle = \frac{1}{\sqrt{\lfloor NR \rceil}} \sum_{i=0}^{\lfloor NR \rceil -1} |f_{I,N}(i)\rangle$, where $I = \sum e_i g_i$.

Claim 5.1. *If I and J are the same element of the class group, then $|\phi_I\rangle$ and $|\phi_J\rangle$ have inner product at least $1 - \sqrt{\Delta}/N$.*

Proof. Consider a segment between ideal J and the next ideal J'. There are at least $N/\sqrt{\Delta}$ rational numbers mapped into this region. As in the analysis in Lemma 3.2 this number is always M or $M + 1$ for some M depending J. Therefore the two vectors have common support of at least a $1 - \sqrt{\Delta}/N$ fraction of basis vectors. □

The rounding problem mentioned in the introduction can be fixed by using the functions $f_{J,j}$. The choice of L must be taken larger by an exponential factor than the total number of reduced ideals, which is at most Δ [Len82b]. For simplicity we list f_J below.

Algorithm 5.1 (Group element to quantum state map).
Input: A set of coefficients $e_1, \ldots, e_k$.
Output: An approximation of the quantum state $|\phi_{e_1,\ldots,e_k}\rangle = \frac{1}{\sqrt{\lfloor NR \rceil}} \sum_{i=0}^{\lfloor NR \rceil -1} |f_I(i)\rangle$, where $I = \sum e_i g_i$.

```
1. Approximate the regulator $R$ using Algorithm 3.1. Let $N = \Delta$.
2. Compute the ideal $I = \sum_i e_i g_i$, resulting in $|I\rangle$.
3. Split into a superposition of distances and evaluate $f_I$ to get $\sum_{i=0}^{\lfloor NR \rceil -1} |I, i, f_I(i)\rangle$.
4. Use Algorithm 4.1 to compute $i/N$ with exponentially high probability.
5. Erase the second register, which contains $i$.
6. Uncompute $i/N$, resulting in a state exponentially close to $\frac{1}{\sqrt{\lfloor NR \rceil}} \sum_{i=0}^{\lfloor NR \rceil -1} |I, f_I(i)\rangle$.
7. Uncompute $I$ from $e_1, \ldots, e_k$, resulting in $\frac{1}{\sqrt{\lfloor NR \rceil}} \sum_{i=0}^{\lfloor NR \rceil -1} |f_I(i)\rangle$.
```

Using this algorithm in the standard hidden subgroup problem algorithm for the function evaluation shows:

Corollary 5.1. *The class group and class number of a real quadratic number field can be found in quantum polynomial-time assuming the GRH.*

6 Open Problems

In this paper we solved some problems from computational algebraic number theory. Given an order $\mathcal{O}$ of a real quadratic number field $\mathbb{Q}(\sqrt{d})$ we showed how to compute the regulator, how to compute the distance of an ideal, and how to compute the class group. For higher dimensional number fields, there are a number of problems still open. In this paper we looked at *quadratic* number fields, which means that K has degree two over $\mathbb{Q}$. To get a number field of degree n a root of a degree n equation is adjoined to $\mathbb{Q}$. The open problems include finding the class group and class number, the group of units and regulator, and solving the principal ideal problem. This list can be found in [Coh93, pg. 217].

We should also point out that there are similar questions for *imaginary* quadratic number fields, which are defined similarly with d being negative. The problems here are much easier however. Finding units is now polynomial time and there are only a finite number. Factoring reduces to computing the class group, but quantum algorithms for computing the class group are automatic from standard quantum techniques, because each group element has a unique efficiently computable representative.

Finally we mention that the following problems for quadratic number fields are in $NP \cap co-NP$ assuming the GRH [BW91] and assuming a constant degree number field: deciding if an ideal is principal, deciding if a set of ideals generate the class group, deciding if a set of ideals are a basis for the class group.

Acknowledgments: Thanks to Hendrik Lenstra for many useful discussions and for suggesting these problems. Also thanks to Kirsten Eisenträger, Richard Jozsa, Ashwin Nayak, Umesh Vazirani, and Ulrich Vollmer for useful discussions.

References

[BTW95] Johannes Buchmann, Christoph Thiel, and Hugh C. Williams. Short representation of quadratic integers. In Wieb Bosma and Alf J. van der Poorten, editors, *Computational Algebra and Number Theory, Sydney 1992*, volume 325 of *Mathematics and its Applications*, pages 159–185. Kluwer Academic Publishers, 1995.

[Buc89] J. Buchmann. A subexponential algorithm for the determination of class groups and regulators of algebraic number fields. In *Séminaire de théorie des nombres*, pages 28–41. Paris, 1989.

[BV97] Ethan Bernstein and Umesh Vazirani. Quantum complexity theory. *SIAM Journal on Computing*, 26(5):1411–1473, October 1997.

[BW89a] Johannes Buchmann and H. C. Williams. On the existence of a short proof for the value of the class number and regulator of a real quadratic field. In *Number theory and applications (Banff, AB, 1988)*, pages 327–345. Kluwer Acad. Publ., Dordrecht, 1989.

[BW89b] Johannes A. Buchmann and Hugh C. Williams. A key exchange system based on real quadratic fields (extended abstract). In G. Brassard, editor, *Advances in Cryptology—CRYPTO '89*, volume 435 of *Lecture Notes in Computer Science*, pages 335–343. Springer-Verlag, 1990, 20–24 August 1989.

[BW91] Johannes Buchmann and Hugh C. Williams. Some remarks concerning the complexity of computing class groups of quadratic fields. *Journal of Complexity*, 7(3):311–315, 1991.

[Coh93] Henri Cohen. *A course in computational algebraic number theory*, volume 138 of *Graduate Texts in Mathematics*. Springer-Verlag, 1993.

[EHK99] Mark Ettinger, Peter Høyer, and Emanuel Knill. Hidden subgroup states are almost orthogonal. Technical report, quant-ph/9901034, 1999.

[FIM$^+$03] Katalin Friedl, Gabor Ivanyos, Frederic Magniez, Miklos Santha, and Pranab Sen. Hidden translation and orbit coset in quantum computing. In *Proceedings of the Thirty-Fifth Annual ACM Symposium on Theory of Computing*, San Diego, CA, 9–11 June 2003.

[GSVV01] Michaelangelo Grigni, Leonard Schulman, Monica Vazirani, and Umesh Vazirani. Quantum mechanical algorithms for the nonabelian hidden subgroup problem. In *Proceedings of the Thirty-Third Annual ACM Symposium on Theory of Computing*, Crete, Greece, 6–8 July 2001.

[Hal02] Sean Hallgren. Polynomial-time quantum algorithms for Pell's equation and the principal ideal problem. In *Proceedings of the Thirty-Fourth Annual ACM Symposium on Theory of Computing*, pages 653–658, Montreal, Quebec, Canada, 19–21 May 2002.

[HRTS00] Sean Hallgren, Alexander Russell, and Amnon Ta-Shma. Normal subgroup reconstruction and quantum computation using group representations. In *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing*, pages 627–635, Portland, Oregon, 21–23 May 2000.

[IMS01] Gábor Ivanyos, Frédéric Magniez, and Miklos Santha. Efficient quantum algorithms for some instances of the non-abelian hidden subgroup problem. In *Proceedings of the Thirteenth Annual ACM Symposium on Parallel Algorithms and Architectures*, pages 263–270, Heraklion, Crete Island, Greece, 4-6 July 2001.

[Joz03] Richard Jozsa. Notes on Hallgren's efficient quantum algorithm for solving Pell's equation. Technical report, quant-ph/0302134, 2003.

[Kup03] Greg Kuperberg. A subexponential-time quantum algorithm for the dihedral hidden subgroup problem. Technical report, quant-ph/0302134, 2003.

[Len82a] Hendrik W. Lenstra, Jr. On the calculation of regulators and class numbers of quadratic fields. In J. V. Armitage, editor, *Journees Arithmetiques, Exeter 1980*, volume 56 of *London Mathematical Society Lecture Notes Series*, pages 123–150. Cambridge University Press, 1982.

[Len82b] H. W. Lenstra, Jr. On the computation of regulators and class numbers of quadratic fields. *Lond. Math. Soc. Lect. Note Ser.*, 56:123–150, 1982.

[Len02] H. W. Lenstra, Jr. Solving the Pell equation. *Notices Amer. Math. Soc.*, 49(2):182–192, February 2002.

[LL93] A.K. Lenstra and H.W. Lenstra, editors. *The Development of the Number Field Sieve*, volume 1544 of *Lecture Notes in Mathematics.* Springer–Verlag, 1993.

[MRRS04] Cris Moore, Daniel Rockmore, Alexander Russell, and Leonard Schulman. The hidden subgroup problem in affine groups: Basis selection in fourier sampling. In *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, New Orleans, LA, 11–13 January 2004.

[NZM91] Ivan Niven, Herbert S. Zuckerman, and Hugh L. Montgomery. *An introduction to the theory of numbers.* John Wiley & Sons Inc., New York, fifth edition, 1991.

[Sch86] A. Schrijver. *Theory of Linear and Integer Programming*, chapter 15.1 : Karmarkar's polynomial–time algorithm for linear programming, pages 190–194. John Wiley & Sons, New York, 1986.

[Sho97] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, October 1997.

[Sho03] Peter W. Shor. Why haven't more quantum algorithms been found? *Journal of the ACM*, 50(1):87–90, January 2003.

[Sim97] Daniel R. Simon. On the power of quantum computation. *SIAM Journal on Computing*, 26(5):1474–1483, October 1997.

[vDHI03] Wim van Dam, Sean Hallgren, and Lawrence Ip. Quantum algorithms for some hidden shift problems. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, Baltimore, MD, 2003.

[Vol00] Ulrich Vollmer. Asymptotically fast discrete logarithms in quadratic number fields. In *Algorithmic Number Theory Symposium IV*, volume 1838, pages 581–594, 2000.

[Wat01] John Watrous. Quantum algorithms for solvable groups. In *Proceedings of the Thirty-Third Annual ACM Symposium on Theory of Computing*, pages 60–67, Crete, Greece, 6–8 July 2001.

[Wil00] H. C. Williams. Solving the pell equation. In *Proc. Millennial Conference on Number Theory*, volume 3, pages 397–435, 2000.