Class group computations in number fields and applications to cryptology

Alexandre Gélin

THÈSE DE DOCTORAT DE
L'UNIVERSITÉ PIERRE ET MARIE CURIE

Spécialité

Informatique

École Doctorale Informatique, Télécommunications et Électronique (Paris)

Présentée par

Alexandre GÉLIN

Pour obtenir le grade de

DOCTEUR de l'UNIVERSITÉ PIERRE ET MARIE CURIE

Calcul de Groupes de Classes d'un Corps de Nombres et Applications à la Cryptologie

Thèse dirigée par Antoine JOUX et Arjen LENSTRA

soutenue le **vendredi 22 septembre 2017**

après avis des **rapporteurs** :

M. Andreas ENGE	Directeur de Recherche, Inria Bordeaux-Sud-Ouest & IMB
M. Claus FIEKER	Professeur, Université de Kaiserslautern

devant le **jury** composé de :

M. Karim BELABAS	Professeur, Université de Bordeaux
M. Andreas ENGE	Directeur de Recherche, Inria Bordeaux-Sud-Ouest & IMB
M. Claus FIEKER	Professeur, Université de Kaiserslautern
M. Louis GOUBIN	Professeur, Université de Versailles-St-Quentin-en-Yvelines
M. Antoine JOUX	Chaire de Cryptologie à la Fondation UPMC, Paris
M. Arjen LENSTRA	Professeur, École Polytechnique Fédérale de Lausanne
Mme Ariane MÉZARD	Professeur, Université Pierre et Marie Curie, Paris
M. Nigel SMART	Professeur, Université de Bristol

Class Group Computations in Number Fields and Applications to Cryptology

Alexandre GÉLIN

Team ALMASTY, Laboratoire d'Informatique de Paris 6

UPMC Sorbonne Universités

Laboratoire d'Informatique de Paris 6
Équipe ALMASTY
UPMC - Sorbonne Universités
4, place Jussieu
75005 Paris, FRANCE

Université Pierre et Marie Curie
École doctorale EDITE – ED 130
Faculté d'ingénierie – UFR 919
4, place Jussieu
75005 Paris, FRANCE

Mathematics may be defined as the subject
where we never know what we are talking about,
nor whether what we are saying is true.
— Bertrand Russell

À tous ceux qui ne pourront lire ces mots…

Abstract

A common way to construct public-key cryptosystems is to use the discrete exponentiation in a finite group where the inverse problem, the discrete logarithm problem, is considered difficult. Several groups are commonly used: multiplicative group of finite fields or Jacobians of algebraic curves defined over finite fields. The class group of a number field is another candidate which was sometimes proposed. To study its viability, it is important to identify the difficulty for this kind of group to determine its cardinality, its structure and the possible hardness of the discrete logarithm problem. Irrespective of cryptologic applications, methods to determine the properties of these groups is of independent mathematical interest.

In this thesis, we focus on class group computations in number fields. We start by describing an algorithm for reducing the size of a defining polynomial of a number field. There exist infinitely many polynomials that define a specific number field, with arbitrarily large coefficients, but our algorithm constructs the one that has the absolutely smallest coefficients. The advantage of knowing such a "small" defining polynomial is that it makes calculations in the number field easier because smaller values are involved. In addition, thanks to such a small polynomial, one can use specific algorithms that are more efficient than the general ones for class group computations.

The generic algorithm to determine the structure of a class group is based on ideal reduction, where ideals are viewed as lattices. We describe and simplify the algorithm presented by Biasse and Fieker in 2014 at ANTS and provide a more thorough complexity analysis for it. We also examine carefully the case of number fields defined by a polynomial with small coefficients. We describe an algorithm similar to the Number Field Sieve, which, depending on the field parameters, may reach the hope for complexity $L\left(\frac{1}{3}\right)$.

Finally, our results can be adapted to solve an associated problem: the Principal Ideal Problem. Given any basis of a principal ideal (generated by a unique element), we are able to find such a generator. As this problem, known to be hard, is the key-point in several homomorphic cryptosystems, the slight modifications of our algorithms provide efficient attacks against these cryptographic schemes.

Résumé

L'une des voies privilégiées pour la construction de cryptosystèmes à clé publique est l'utilisation dans des groupes finis de l'exponentiation discrète, dont le problème inverse, celui du logarithme discret, est réputé difficile. Plusieurs groupes sont classiquement utilisés : le groupe multiplicatif d'un corps fini ou la Jacobienne d'une courbe algébrique définie sur un corps fini. Le groupe de classes d'un corps de nombres est un autre candidat qui a parfois été proposé. Pour étudier sa viabilité, il est important de bien cerner la difficulté de déterminer la cardinalité et la structure du groupe, ainsi que la difficulté éventuelle du problème du logarithme discret. Indépendamment de ces applications cryptographiques, les méthodes utilisées pour résoudre ces problèmes ont aussi un intérêt mathématique.

Dans cette thèse, nous nous intéressons au calcul du groupe de classes d'un corps de nombres. Nous débutons par décrire un algorithme de réduction du polynôme de définition d'un corps de nombres. Il existe une infinité de polynômes qui définissent un corps de nombres fixé, avec des coefficients arbitrairement gros. Notre algorithme calcule celui qui a les plus petits coefficients. L'avantage de connaître un petit polynôme de définition est qu'il simplifie les calculs entre éléments de ce corps de nombres, en impliquant des quantités plus petites. En outre, la connaissance d'un tel polynôme permet l'utilisation d'algorithmes plus efficaces que dans le cas général pour calculer le groupe de classes.

L'algorithme général pour calculer la structure du groupe de classes repose sur la réduction d'idéaux, vus comme des réseaux. Nous décrivons et simplifions l'algorithme présenté par Biasse et Fieker en 2014 à ANTS et approfondissons l'analyse de complexité. Nous nous sommes aussi intéressés au cas des corps de nombres définis par un polynôme à petits coefficients. Nous décrivons un algorithme similaire au crible par corps de nombres (NFS) dont la complexité en fonction des paramètres du corps de nombres peut atteindre $L\left(\frac{1}{3}\right)$.

Enfin, nos algorithmes peuvent être adaptés pour résoudre un problème lié : le Problème de l'Idéal Principal. Étant donné n'importe quelle base d'un idéal principal (généré par un seul élément), nous sommes capables de retrouver ce générateur. Cette application de nos algorithmes fournit une attaque efficace contre certains schémas de chiffrement homomorphe basés sur ce problème.

Contents

Introduction

1 Motivation

Cryptographie. La cryptographie moderne se divise en deux grandes sous-familles. La plus ancienne est la *cryptographie symétrique*. L'idée principale est que l'émetteur et le destinataire d'un message partagent un secret commun, appelé *clé secrète*. En 1976, Diffie et Hellman introduisent la *cryptographie asymétrique* — aussi appelée *cryptographie à clé publique* — dans leur article intitulé *New Directions in Cryptography* [DH76]. L'avancée majeure est que rien ne doit être partagé par l'émetteur et le destinataire : ce dernier génère un couple `(clé privée, clé publique)`, garde la `clé privée` secrète et rend publique la `clé publique`. Alors l'émetteur est capable de chiffrer un message grâce à la `clé publique`, et seul le destinataire peut le déchiffrer, grâce à sa `clé privée`.

La cryptographie asymétrique repose sur des problèmes calculatoires difficiles. Elle n'est donc utilisée en général que pour le transfert d'une clé secrète. Cette clé secrète sera alors utilisée dans un schéma symétrique pour chiffrer une séquence de messages plus longs. La cryptographie symétrique repose sur des algorithmes plus simples et plus rapides.

La cryptographie est utilisée pour différents aspects de la sécurité. Le principal — et historiquement le premier — est la *confidentialité*. Mais elle peut aussi fournir des résultats d'*intégrité*, en détectant une altération des données transmises, et d'*authentification*, pour assurer l'identité de l'émetteur. Dans ce sens, les primitives cryptographiques peuvent être utilisées pour chiffrer/déchiffrer, mais aussi adaptées pour en faire des *algorithmes de signatures électroniques*. Nous ne nous sommes pas intéressés à ces aspects pratiques dans cette thèse : nous considérons uniquement les problèmes mathématiques sous-jacents aux schémas que nous étudions.

Cryptographie asymétrique. Une analogie simple pour expliquer le chiffrement asymétrique est celle d'une boîte à lettres. Celle-ci est accessible à n'importe qui et sa localisation (l'adresse postale) est ce qui correspond à la clé publique. N'importe qui connaissant l'adresse

postale peut aller y déposer une lettre. Au contraire, seule la personne qui possède la clé de cette boîte à lettres peut l'ouvrir et récupérer la lettre.

Les primitives asymétriques reposent principalement sur des fonctions *à sens unique* : faciles à calculer pour n'importe quelle entrée, mais difficiles à inverser en connaissant l'image d'une entrée aléatoire. Les mots *"facile"* et *"difficile"* doivent être ici compris au sens calculatoire du terme, c'est-à-dire qu'il existe un algorithme en temps polynomial pour un sens, mais pas pour son inverse. Pour revenir à l'analogie de la boite à lettres, il est facile de glisser une enveloppe dans la boîte alors qu'il est quasiment impossible de l'en faire ressortir sans l'ouvrir. Plusieurs candidats existent pour ces fonctions à sens unique. Nul ne sait vraiment si ces fonctions sont effectivement à sens-unique, mais elles sont communément considérées comme telles, au regard des recherches infructueuses effectuées jusqu'ici pour trouver un algorithme inverse efficace.

Le premier *problème difficile* que nous mettons en avant est celui de la factorisation des entiers. Soient p et q deux grands nombres premiers. Il est facile d'en calculer le produit $n = pq$, alors qu'à l'opposé, étant donné n, il est très difficile de retrouver p et q. Ce problème est à la base du premier schéma de chiffrement à clé publique RSA, introduit en 1978 par Rivest, Shamir et Adleman [RSA78] et toujours très répandu de nos jours. Un autre exemple de schéma basé sur le problème de la factorisation des entiers a été introduit par Paillier en 1999 [Pai99].

Le second problème difficile que nous introduisons est celui du logarithme discret (DLP) : étant donné un groupe cyclique G dont g est un générateur et un élément aléatoire $h \in G$, trouver un exposant $x \in \mathbf{Z}$ tel que $h = g^x$. La cryptographie basée sur le problème du logarithme discret est très répandue de nos jours et les groupes sont devenus une structure essentielle en cryptographie. Les premières descriptions de ce problème utilisaient le groupe multiplicatif d'un corps fini. Ce fut le cas par exemple de l'échange de clés Diffie-Hellman [DH76] et du chiffrement ElGamal [ElG84, ElG85]. Cependant, ces corps possèdent tellement de structure que les tailles requises pour des applications cryptographiques sont relativement grandes.

En 1985, Koblitz [Kob87] et Miller [Mil85] ont indépendamment proposé l'utilisation des *courbes elliptiques* en cryptographie. Le premier avantage de ces courbes est une réduction de la taille des clés, ce qui limite l'espace de stockage requis et les transmissions nécessaires. Par exemple, une clé publique de 256 bits pour une courbe elliptique apporte une sécurité comparable à celle d'une clé publique de 3072 bits pour un chiffrement RSA. La cryptographie à base de courbes elliptiques a gagné en popularité ces dix dernières années et a ensuite été étendue aux Jacobiennes de courbes de genre supérieur.

Chiffrement Homomorphe. Le chiffrement homomorphe est une forme de chiffrement qui permet d'effectuer des opérations directement sur les textes chiffrés, générant ainsi un résultat qui, une fois déchiffré, est égal au résultat des mêmes opérations sur les textes clairs. Mener ces calculs sur des données chiffrées permet de traiter des données sensibles sans révéler aucune information sur le propriétaire. Une application possible serait par exemple de calculer à distance dans le *Cloud*. RSA et ElGamal sont deux exemples de schémas de chiffrement multiplicativement homomorphe, alors que Paillier est un schéma de chiffrement additivement homomorphe.

Un cryptosystème qui permet de réaliser n'importe quelle séquence d'opérations est dit *complètement homomorphe* (FHE) et est beaucoup plus puissant. La proposition présentée par Boneh, Goh et Nissim dans [BGN05] a été la première à permettre à la fois additions et multiplications. Cette solution a tout de même un inconvénient, puisqu'une seule multiplication est possible. Ce type de schéma est dit *partiellement homomorphe*.

Gentry décrit en 2009 la première construction complètement homomorphe, en utilisant de la cryptographie basée sur les réseaux [Gen09]. Son schéma, inspiré de NTRU [HPS98], permet de mixer à la fois additions et multiplications sur les textes chiffrés, à partir desquelles il est possible de réaliser n'importe quel calcul. Ce résultat important a entrainé la présentation de nouvelles techniques. Les plus connues ont été proposées par Brakerski, Gentry et Vaikuntanathan [BGV12], Fan et Vercauteren [FV12], Bos, Lauter, Loftus et Naehrig (YASHE) [BLLN13] et Khedr, Gulak et Vaikuntanathan (SHIELD) [KGV16].

Cryptographie Post-Quantique. L'ordinateur quantique pourrait un jour devenir une réalité. Quand ce jour viendra, la sécurité de plusieurs primitives utilisées de nos jours — comme RSA ou le DLP — s'effondrera et elles deviendront obsolètes. En effet, il existe un algorithme quantique, l'*algorithme de Shor* [Sho97], qui est capable de résoudre ces problèmes en temps polynomial, sur un ordinateur quantique suffisamment puissant.

Heureusement, un ordinateur quantique d'une telle puissance n'existe pas. Mais certains experts — pas tous — s'accordent sur le fait, qu'un jour, dans le futur, un tel ordinateur existera. La cryptographie post-quantique comprend tous les schémas considérés comme sûrs contre les attaques menées par un ordinateur quantique. Elle est différente de la *cryptographie quantique*, qui regroupe les schémas qui font l'usage de phénomènes quantiques.

Plusieurs problèmes difficiles ont été proposés pour la cryptographie post-quantique, et les solutions les plus étudiées peuvent être divisées en quatre familles :

- la cryptographie basée sur les réseaux : NTRU [HPS98], (R)LWE [Reg05, Pei09, LPR13], GGH [GGH13]
- la cryptographie basée sur les codes correcteurs : McEliece [McE78]
- la cryptographie multivariée : C* [MI88], UOV [KPG99]
- la cryptographie basée sur les isogénies entre courbes elliptiques supersingulières : SIDH [JF11, FJP14]

Les cryptosystèmes qui nous intéressent dans cette thèse appartiennent à la catégorie des schémas basés sur les réseaux, et plus précisément des réseaux particuliers : les *réseaux-idéaux*. Parallèlement, nous évoquons SIDH un peu plus loin dans cette introduction.

Théorie Algébrique des Nombres. Grâce à l'avènement du chiffrement totalement homomorphe et de la cryptographie post-quantique, la Théorie Algébrique des Nombres devient de plus en plus populaire en cryptologie. En effet, l'utilisation des réseaux a permis de construire des cryptosystèmes totalement homomorphe et résistant aux attaques quantiques. Parmi les réseaux les plus utilisés en cryptographie apparaissent les *réseaux-idéaux*, des réseaux avec une structure additionnelle, comme ils proviennent d'un idéal dans un corps de nombres. Ce choix est motivé par le fait que leur utilisation permet de réduire à la fois le stockage nécessaire et les temps de calculs. Par exemple, Smart et Vercauteren [SV10] ont proposé une instanciation pratique du résultat théorique de Gentry [Gen09], dans les corps cyclotomiques de dimension une puissance de deux. La sécurité de ce schéma repose sur la difficulté de trouver un *petit* générateur d'un idéal principal (un idéal généré par un unique élément). Les solutions à ce problème font naturellement appel aux méthodes utilisées pour calculer le groupe de classes d'un corps de nombres, en raison de la nature des structures qui sont impliquées.

Parallèlement, les groupes de classes ont déjà une histoire avec la cryptographie. En 1988, Buchmann et Williams [BW88] ont proposé une variante de l'échange de clés Diffie-Hellman dans des groupes de classes de corps quadratiques imaginaires. En effet, en tant que groupes finis, ils sont des candidats légitimes au problème du logarithme discret. La différence majeure dans ce cas est que l'ordre du groupe est inconnu, ce qui n'est pas un problème insurmontable. Il existe aussi une variante à DSA (Algorithme de Signature Digitale) [Sch91] pour les signatures, appelée RDSA — le "R" vient du fait que le schéma de signature est basé sur la difficulté de trouver la racine e-ième d'un élément. Un autre schéma, NICE [PT00], utilise les idéaux d'un corps quadratique imaginaire. Le secret est un nombre premier p et la clé publique correspond

à un idéal dans l'ordre d'indice p. La sécurité repose donc essentiellement sur la difficulté de factoriser le discriminant de l'ordre, qui est de la forme $p^2 q$, pour un nombre premier q.

Tous ces cryptosystèmes ont d'abord été décrits pour un corps quadratique imaginaire. Une raison est que ces corps sont connus pour avoir un *gros* groupe de classes — de la taille de la racine carré du discriminant du corps. Il existe des généralisations au cas des corps quadratiques réels, où les unités jouent en général le rôle principal (voir [BBT94, SBW94] pour plus de détails). Finalement, à l'orée des années 2000, sont apparues dans la littérature des généralisations en degré arbitraire. En plus de la hausse des temps de calcul liée à la dimension élevée, les applications requièrent la connaissance de corps de nombres ayant un gros groupe de classes. De tels résultats ont été fournis par Stender [Ste75] pour les degrés 3, 4 et 6, Buchmann [Buc87] pour le degré 4, Emma Lehmer [Leh88] pour le degré 5, etc.

Groupes de classes. Une réponse légitime à la question *"Pourquoi calculer des groupes de classes ?"* est *"Parce que Gauss l'a fait."*. En effet, calculer des groupes de classes est un problème mathématique intéressant qui mérite d'être un but à lui tout seul. Leur étude a déjà mené à des découvertes majeures en théorie algorithmique des nombres. Le résultat le plus célèbre est sans nul doute l'algorithme *Pas de bébés – Pas de géants*, introduit en 1969 par Shanks [Sha69]. Cet algorithme est connu pour être asymptotiquement optimal pour la résolution du DLP dans un groupe générique (sans structure particulière). Et il a été découvert dans le cadre du calcul du groupe de classes ! En effet, Shanks décrit à l'origine cette méthode pour calculer le nombre de classes d'un corps quadratique imaginaire. Il existe aussi des algorithmes pour la factorisation d'entiers basés sur le calcul du groupe — ou au moins d'un sous-groupe — de classes d'un corps quadratique (voir [Coh93, Sections 8.6 & 10.2]).

Les groupes de classes ont aussi eu un impact marquant dans une des premières preuves partielles du Grand Théorème de Fermat :

Théorème. *Il n'existe pas d'entiers non-nuls x, y et z tels que* $x^n + y^n = z^n$ *dès que* $n \geq 3$.

La preuve fournie par Fermat couvre le cas $n = 4$ et montre qu'il suffit, pour résoudre le problème, de s'intéresser à l'équation $x^p + y^p = z^p$, avec p premier et x, y, z deux-à-deux premiers entre eux. La première preuve générique nous vient de Sophie Germain qui prouva le cas où x, y et z ne sont pas des multiples de p, pour tout premier p tel que $2p+1$ soit aussi premier : les *nombres premiers de Sophie Germain*. Avant la preuve totale apportée par Wiles [Wil95] en 1994 — qui utilise des courbes elliptiques — de nombreuses preuves ont été proposées. En 1848, Lamé utilise les *racines de l'unité*, nombres complexes vérifiant $\zeta^p = 1$, pour décomposer

$$x^p + y^p = (x+y)(x+\zeta y)(x+\zeta^2 y)\cdots(x+\zeta^{p-1} y). \tag{1}$$

Néanmoins, sa preuve n'est pas valide, puisqu'elle suppose l'unicité de la factorisation dans un corps cyclotomique. Kummer introduit alors une notion proche des idéaux pour obtenir le résultat de décomposition unique en idéaux premiers. Cela permit de compléter la preuve de Lamé pour l'ensemble des *premiers réguliers*.

Définition. Un nombre premier est dit *régulier* si l'ordre du groupes de classes de $\mathbf{Q}(\zeta)$ n'est pas un multiple de p, pour $\zeta \neq 1$ tel que $\zeta^p = 1$.

En combinant l'Équation (1) et $x^p + y^p = z^p$, il apparaît que chaque idéal généré par $(x+\zeta^k y)$, pour $1 \leq k \leq p-1$, est une puissance p-ième d'un idéal $\mathfrak{a}_k$. Pour un premier régulier p, comme p ne divise pas l'ordre du groupe de classes, le fait que l'idéal $\mathfrak{a}_k^p$ soit principal entraîne que l'idéal $\mathfrak{a}_k$ est aussi principal. Ce point est le point crucial de la preuve, le reste ne consiste qu'à réécrire $x+\zeta y$ de sorte qu'une contradiction apparaisse.

Nous ne savons toujours pas s'il existe ou non une infinité de nombres premiers réguliers. Jensen a prouvé en 1915 qu'une infinité de premiers ne sont pas réguliers. Cependant, il est communément admis que les premiers réguliers ont une densité $e^{-\frac{1}{2}}$ asymptotiquement, ce qui représente environ 60% des premiers (il s'agit d'une conjecture de Siegel datant de 1964 qui n'a toujours pas été prouvée). Par exemple, Kummer avait identifié que les premiers irréguliers plus petits que 100 sont 37, 59 et 67.

2 Organisation de la thèse et résultats

Le but de cette thèse est d'étudier et d'améliorer le calcul du groupe de classes mais aussi d'aborder certaines applications à la cryptologie.

Nous présentons dans le Chapitre 1 les outils mathématiques nécessaires à la bonne compréhension de la suite du manuscrit. Nous donnons par exemple les définitions du groupe de classes et des unités d'un corps de nombres, afin de se familiariser avec les objets que nous cherchons à calculer. Un rappel succinct sur les réseaux euclidiens est aussi inclus, puisqu'ils représentent une partie essentielle de notre travail.

Le Chapitre 2 est consacré à l'historique du groupe de classes, depuis Gauss jusqu'à l'état de l'art actuel. C'est l'occasion d'introduire des méthodes connues, parfois associées à des problèmes différents qui ont été plus étudiés que le groupe de classes. Nous décrivons le chemin allant des corps quadratiques jusqu'aux corps de nombres de degrés arbitraires.

Dans le Chapitre 3, nous décrivons un algorithme pour trouver un polynôme de définition d'un corps de nombres avec des petits coefficients. En effet, il existe une infinité de polynômes

de définition pour un corps de nombres fixé, mais utiliser celui qui a les coefficients les plus petits possible est généralement le meilleur choix. Nous présentons l'algorithme ainsi que l'analyse de sa complexité et étudions son impact dans le cadre du calcul du groupe de classes. Nous remarquons qu'en le considérant comme un pré-calcul, il permet d'étendre un résultat de Biasse et Fieker [BF14] a de plus grandes classes de corps de nombres (de petits degrés).

Dans le Chapitre 4, nous nous intéressons à l'algorithme générique pour le calcul du groupe de classes. Cet algorithme est utilisé lorsque nous ne connaissons pas de polynôme de définition à petits coefficients, ou qu'il est trop coûteux d'en chercher un. Inspiré d'un résultat de Biasse et Fieker [BF14], nous donnons une version simplifiée de leur algorithme et améliorons leur analyse de complexité afin d'identifier les paramètres optimaux pour réduire le temps de calcul.

Dans le Chapitre 5, nous nous concentrons sur les améliorations possibles lorsqu'un polynôme de définition à petits coefficients est connu. Dans ce cas, nous sommes capables de proposer un algorithme dont le temps de calcul est considérablement réduit par rapport à l'algorithme générique. Nous donnons aussi un résultat pour un problème lié, le Problème de l'Idéal Principal (PIP) : étant donné une base d'un idéal principal, c'est-à-dire généré par un unique élément, il est difficile d'en trouver un générateur.

Finalement, dans le Chapitre 6, la cryptanalyse d'un schéma de chiffrement totalement homomorphe est présentée. Cette attaque se base sur la résolution du problème de l'idéal principal pour un petit générateur, dans un corps cyclotomique d'indice une puissance d'un nombre premier. Ce cryptosystème a été décrit par Smart et Vercauteren à PKC en 2010 [SV10] comme une instanciation pratique du schéma FHE de Gentry [Gen09]. Nous tirons profit de la structure de corps cyclotomique pour réduire la dimension du problème puis utilisons les techniques décrites au Chapitre 5 pour retrouver un générateur. La dernière étape, qui consiste à dériver de ce générateur arbitraire un petit générateur, a déjà été résolue par Cramer, Ducas, Peikert et Regev et présentée à EUROCRYPT en 2016 [CDPR16].

Publications

Les résultats que nous avons obtenus durant cette thèse ont mené à la publication d'articles dans des conférences et journaux internationaux. Ce manuscrit ne reprend que les quatre premiers de la liste suivante, puisqu'ils sont ceux qui concernent le calcul du groupe de classes. Ils correspondent respectivement aux Chapitres 3, 4, 5 et 6. Nous discutons brièvement les deux autres dans la section suivante.

[GJ16] **Reducing number field defining polynomials : an application to class group computations**
avec Antoine Joux, *LMS Journal of Computation and Mathematics & ANTS XII, 2016.*

[GJ17a] **On the complexity of class group computations for large-degree number fields**
avec Antoine Joux, *En cours de soumission.*

[GJ17b] **Reducing the complexity for class group computations using small defining polynomials**
avec Antoine Joux, *En cours de soumission.*

[BEF+17] **Computing generator in cyclotomic integer rings — A subfield algorithm for the Principal Ideal Problem in L(1/2) and application to the cryptanalysis of a FHE scheme**
avec Jean-François Biasse, Thomas Espitau, Pierre-Alain Fouque et Paul Kirchner, *EUROCRYPT 2017.*

[GKL17] **Parametrizations for families of ECM-friendly curves**
avec Thorsten Kleinjung et Arjen K. Lenstra, *ISSAC 2017.*

[GW17] **Loop-abort faults on supersingular isogeny cryptosystems**
avec Benjamin Wesolowski, *PQCrypto 2017.*

Résultats complémentaires

ECM. L'algorithme de factorisation d'entiers par les courbes elliptiques (ECM) a été introduit par H.W. Lenstra en 1985 et publié deux ans plus tard dans [Len87]. Cette méthode est asymptotiquement la plus rapide pour trouver des facteurs relativement petits d'un gros entier composé. Bien que le crible par corps de nombres [LL93] soit l'algorithme le plus efficace génériquement pour la factorisation, il existe deux cas d'usage classique pour ECM : il est communément utilisé pour trouver des facteurs d'un gros entier lorsque la taille des facteurs premiers n'est pas connue (Propper a trouvé le plus gros jusqu'ici, un facteur de 274 bits de $7^{337}+1$) et il est utilisé dans l'étape de *co-factorisation* du crible par corps de nombres (où beaucoup de nombres composés de taille relativement petite doivent être factorisés).

Étant donné un nombre composé impair N à factoriser, ECM réalise des opérations arithmétiques sur des courbes elliptiques, que nous considérons comme définies sur le corps fini $\mathbf{F}_p$ de cardinal p, pour un diviseur premier inconnu p de N. L'algorithme identifie p si le cardinal d'au moins une de ces courbes définies sur $\mathbf{F}_p$ est lisse, c'est-à-dire qu'il est le produit de petits nombres premiers. Pour cette raison, les courbes utilisées sont connues pour avoir des propriétés de lissité favorables, comme par exemple un gros groupe de torsion sur $\mathbf{Q}$ ou un cardinal divisible par un facteur fixé. Des constructions de courbes adaptées à ECM

ont été publiées par Suyama [Suy85] (avec une légère amélioration par Montgomery [Mon87, Section 10.3.2]), Atkin-Morain [AM93] et généralisées par Bernstein *et al.* dans [BBL10].

Bien que l'algorithme ait été initialement formalisé en utilisant des courbes de Weierstrass dans [Len87], jusqu'en 2008 les implémentations utilisaient généralement l'approche de Montgomery décrite dans [Mon87]. Avec l'introduction des courbes d'Edwards [Edw07] dont les propriétés ont été étudiées par Bernstein *et al.* [BBJ+08, BBLP13], les courbes d'Edwards *tordues* avec $a=-1$ et un groupe de torsion isomorphe à $\mathbf{Z}/2\mathbf{Z}\times\mathbf{Z}/4\mathbf{Z}$ sont parmi les courbes les plus adaptées à l'algorithme ECM, comme le montre Berstein *et al.* dans [BBL10]. Pour ces courbes, Barbulescu *et al.* [BBB+12] ont identifié trois familles ayant une probabilité plus élevée d'avoir un cardinal lisse et une quatrième famille encore meilleure. Ils fournissent aussi une paramétrisation pour l'une des trois familles équivalentes, les autres étant seulement illustrées par un nombre fini de courbes construites à la main. En particulier, une paramétrisation pour la quatrième et meilleure famille n'avait pas été trouvée. Par *paramétrisation*, nous entendons la donnée d'une courbe et d'un point de non-torsion sur cette courbe, construite à partir d'une fonction dont l'entrée peut être un point sur une autre courbe elliptique ou un rationnel, nous fournissant alors des paramétrisations *elliptiques* ou *rationnelles*.

Nous avons amélioré les résultats de [BBB+12] en fournissant six nouvelles paramétrisations rationnelles puis utilisé trois d'entre elles pour formaliser cinq nouvelles paramétrisations elliptiques qui nous permettent de générer rapidement des courbes pour chacune des quatre familles identifiées dans [BBB+12]. Nous avons mené les mêmes tests que [BBL10] pour la famille qui, étant donné ses propriétés galoisiennes, est la plus prometteuse. Selon les critères pratiques de [BBL10], l'utilisation de cette nouvelle famille apporte des gains de l'ordre de 1 à 2% comparés aux meilleures courbes connues jusqu'ici, avec un écart extrêmement faible entre les courbes de cette même famille. Cela montre que notre nouvelle paramétrisation devrait apporter un gain pratique dans l'étape de co-factorisation du crible par corps de nombres.

SIDH. Les schémas cryptographiques basés sur les isogénies entre courbes elliptiques supersingulières sont introduits par Charles, Lauter et Goren dans [CLG09], qui étudient la difficulté de trouver un chemin dans les graphes d'isogénies des courbes supersingulières et fournissent comme application une fonction de hachage cryptographique. Ce problème a depuis été utilisé pour d'autres systèmes tels que l'échange de clés et le chiffrement par Jao et De Feo [JF11]. Plusieurs autres primitives basées sur les isogénies supersingulières ont suivi, comme les preuves de connaissance à divulgation nulle de connaissance ou les signatures [FJP14, GPS16, JS14, STW12]. Des implémentations efficaces de ces primitives ont

aussi été publiées : en logiciel [CLN16, SIDH], en matériel [KAKJ17] et en systèmes embarqués [KJA+16].

Si la version basique de l'échange de clés utilise des clés éphémères (comme c'est le cas pour Diffie-Hellman), d'autres schémas nécessitent l'utilisation d'une clé statique pour au moins un des participants. Ces secrets *statiques* constituent la matière première pour les attaques dites *actives*, et une telle attaque a déjà été décrite dans [GPST16]. Elle permet de retrouver les n bits de la clé secrète en environ n interactions avec la victime. Cette attaque peut être bloquée grâce à la méthode de validation de Kirkwood *et al.* [KLM+15], qui n'est en fait qu'une transformée de Fujisaki–Okamoto [FO99] appliquée au contexte des isogénies supersingulières.

Les résultats de [KJA+16], ajoutés au fait que les primitives basées sur les isogénies supersingulières possèdent des tailles de clés bien inférieures à celles des autres candidats au post-quantique, suggèrent que ces primitives sont particulièrement adaptées aux systèmes embarqués. Cela ouvre la porte à de potentielles attaques par canaux auxiliaires.

Nous avons décrit la première attaque par canaux auxiliaires contre les primitives basées sur les isogénies supersingulières. Cette attaque exploite l'injection d'une faute de type *sortie prématurée de boucle*, précédemment introduite pour la cryptographie basée sur les couplages. La structure itérative du calcul des isogénies rend le schéma vulnérable aux attaques par *sortie prématurée*, permettant à un attaquant de retrouver les n bits de la clé en $O(n)$ interactions avec le jeton, pour une complexité négligeable. Aucune des contremesures introduites pour les cryptosystèmes basés sur les isogénies ne permet de se prémunir de cette attaque. Les fautes par *sortie prématurée* ont été prouvées applicables en pratique [BGG+14] et doivent donc être considérées comme une menace sérieuse lorsque ces schémas sont implémentés dans un contexte vulnérable aux attaques physiques.

Introduction

1 Motivation

Cryptography. Modern cryptography can be divided into two main areas. The oldest one is *Symmetric-Key Cryptography*. Symmetric primitives require both the sender and the recipient of a message to share a common secret, called the *secret key*. In 1976, Diffie and Hellman proposed *Public-Key Cryptography* in their article entitled *New Directions in Cryptography* [DH76]. The breakthrough is that nothing needs to be shared between the sender and the recipient: the recipient generates a pair `(private-key, public-key)`, keeps secret the `private-key`, and releases the `public-key`. Then the sender is able to encrypt the message using the `public-key` so that only the recipient can decrypt it, thanks to the `private-key`.

Public-key cryptography — also called *asymmetric cryptography* — is computationally more complex so it is generally used only for the transfer of a symmetric encryption key. This symmetric key is then used to encrypt the rest of the potentially long sequence of messages. Symmetric cryptography is based on more *ad hoc* algorithms and is much faster.

Cryptography is used to tackle different aspects of security. The main one — and historically the first — is certainly the *confidentiality*. However, cryptography also provides *data integrity*, which detects any alteration of the data, and *authentication*, which ensures to the recipient the sender's identity. Therefore, *cryptographic primitives* can be used for encryption/decryption, but also adapted for *digital-signature schemes*. We do not focus on these practical aspects in this thesis: we only consider the underlying mathematical problems of the schemes we look at.

Public-Key Cryptography. An analogy to public-key encryption is that of a locked mailbox with a mail slot. The mail slot is exposed and accessible to the public — its location (the street address) is, in essence, the public key. Anyone knowing the street address can go to the door and drop a written message through the slot. However, only the person who possesses the key can open the mailbox and read the message.

Asymmetric primitives are essentially based on *one-way functions*, functions that are *easy* to compute on every input, but *hard* to invert, given the image of a random input. Here, *"easy"* and *"hard"* are to be understood in the sense of *computational complexity theory*, which means that there exists a polynomial-time algorithm for one operation, but not for the inverse. Back to the analogy with the mailbox, it is easy to put an envelope in the mailbox, whereas the difficult part is to remove it without opening the mailbox. There exist several candidates for one-way functions. It is not known whether these functions are indeed one-way, but we commonly rely on the fact that extensive research has so far failed to produce an efficient inverting algorithm.

The first *hard problem* we want to describe is the integer factorization problem. Let p and q be two large prime numbers. It is easy to multiply them and find $n = pq$. However, given n, it is very hard to recover p and q. This is the foundation of the first public-key encryption scheme RSA, introduced in 1978 by Rivest, Shamir, and Adleman [RSA78], and widely used today. Another scheme based on factorization was introduced by Paillier in 1999 [Pai99].

The second problem we introduce is the *Discrete Logarithm Problem* (DLP): given a cyclic group G together with a generator g and a random element $h \in G$, find an exponent $x \in \mathbf{Z}$ such that $h = g^x$. Cryptography based on the DLP is very widespread so that suitable groups have become an essential structure in cryptography. The first descriptions of this problem used a multiplicative group of a finite field. It was the case for Diffie-Hellman key exchange [DH76] and ElGamal encryption [ElG84, ElG85]. However, these fields have so much structure that their size must be very large to supply sufficient security. In 1985, Koblitz [Kob87] and Miller [Mil85] independently proposed the usage of *elliptic curves* in cryptography. The primary benefit promised by elliptic curve cryptography is a smaller key size, reducing storage and transmission requirements. For instance, a 256-bit elliptic curve public key should provide security comparable to a 3072-bit RSA public key. Elliptic curve cryptography has gained in popularity the last ten years and has been further extended to Jacobians of higher-genus curves.

Homomorphic Encryption. Homomorphic encryption is a form of encryption that allows computations to be carried out on ciphertexts, thus generating an encrypted result which, when decrypted, matches the result of the same operations performed on the plaintexts. Performing computations on the encrypted data provides a way to share confidential information without threatening the privacy of the owner, for instance in the context of *cloud computing*. RSA and ElGamal are two examples of multiplicative homomorphic schemes, while Paillier's cryptosystem is an additive homomorphic scheme.

A cryptosystem that supports arbitrary computations on ciphertexts is known as *Fully Homomorphic Encryption* (FHE) and is far more powerful. The cryptosystem devised by

Boneh, Goh, and Nissim in [BGN05] was the first to allow both additions and multiplications. There is a catch, however: only one multiplication is permitted. The system is thus called *somewhat homomorphic*. Gentry, using lattice-based cryptography, described in 2009 the first construction for a fully homomorphic encryption scheme [Gen09]. Gentry's scheme, based on NTRU [HPS98], supports both addition and multiplication operations on ciphertexts, from which it is possible to construct circuits for performing any arbitrary computation. From this result several new techniques followed. The best known were proposed by Brakerski, Gentry, and Vaikuntanathan [BGV12], Fan and Vercauteren [FV12], Bos, Lauter, Loftus, and Naehrig (YASHE) [BLLN13], and Khedr, Gulak, and Vaikuntanathan (SHIELD) [KGV16].

Post-Quantum Cryptography. One day, quantum computers might become a reality. When that day comes, RSA, the DLP, and other fundamental cryptographic primitives would become obsolete. Indeed, these problems can be easily solved on a sufficiently powerful quantum computer running Shor's algorithm [Sho97]. Luckily, such large quantum computers do not exist yet. Still, some — but not all — experts agree that at one point in the future, they will exist. *Post-Quantum Cryptography* refers to cryptographic primitives that are thought to be secure against an attack by a quantum computer. It is distinct from *Quantum Cryptography*, which refers to cryptosystems that use quantum phenomena.

Many hard problems have been proposed for post-quantum cryptography, but the most promising solutions can be grouped in four families:

- Lattice-based cryptography: NTRU [HPS98], (R)LWE [Reg05, Pei09, LPR13], GGH [GGH13]
- Code-based cryptography: McEliece [McE78]
- Multivariate cryptography: C* [MI88], UOV [KPG99]
- Supersingular elliptic curve cryptography: SIDH [JF11, FJP14]

The cryptosystems we are interested in throughout this thesis belong to lattice-based cryptography, and more precisely for special lattices: *ideal-lattices*. Still, we briefly address SIDH later in this introduction.

Algebraic Number Theory. With the growing interest in Fully Homomorphic Encryption and Post-Quantum Cryptography, Algebraic Number Theory gains in popularity. Indeed, using lattices allows building cryptosystems that are fully homomorphic and quantum-resistant. A non-negligible part of lattice-based cryptography relies on *ideal-lattices*; these are lattices with additional structure, as they come from an ideal in a number field. This choice allows more efficiency in the storage and the runtime of the algorithms. As an example, Smart and

Vercauteren provide in [SV10] a practical instantiation of the theoretical result described by Gentry in [Gen09], in the case of power-of-two cyclotomic fields. The foundation of this scheme relies on the hardness to recover a *small* generator of a *principal ideal* (an ideal generated by only one element). Solutions for this problem naturally involve methods closely linked with class group computations, because of the building blocks used in that primitive.

Class groups also have a history with cryptography. Buchmann and Williams [BW88] proposed in 1988 a variant of the Diffie-Hellman key exchange protocol in class groups of imaginary quadratic number fields. Indeed, as finite groups, class groups are candidates for the DLP. The major difference is that the group order is unknown, but this does not seem to be a problem. There also exists a variant of DSA (Digital Signature Algorithm) [Sch91] for signatures, called RDSA [HM00] — the "R" comes from the fact that the signature scheme is based on the *root problem*, *i.e.,* finding an e-th root of an element. Another scheme, NICE (New Ideal Coset Encryption) [PT00], uses ideals in imaginary quadratic number fields. The secret is a prime number p and the public key corresponds to an ideal in the order of index p. Therefore, the security essentially relies on the difficulty to factor the discriminant of the order, which is of the form p^2q, for a prime number q.

All these cryptosystems were first described in the context of imaginary quadratic number fields. A reason is that those fields are known to have large class groups — the size of the square root of the discriminant of the field. Even though, there exist extensions to real quadratic number fields, where most of the time units play an important role (see [BBT94, SBW94] for more details). Finally, in the 2000s, generalizations to arbitrary-degree number fields appeared in the literature [BBHM02, MNP01]. Besides the growth in the runtime due to the dimension, practical application requires to actually identify families of number fields with large class groups. This was done by Stender [Ste75] for degrees 3, 4, and 6, Buchmann [Buc87] for degree 4, Emma Lehmer [Leh88] for degree 5, etc.

Class groups for mathematical purposes. A very consistent answer to the question *"Why do we compute class groups ?"* is *"Because Gauss did."*. Indeed, computing class groups is an interesting mathematical problem that deserves to be a goal in itself, and their study has led to important breakthroughs in algorithmic number theory. The most popular result is maybe the Baby-Step–Giant-Step algorithm, introduced by Shanks in 1969 [Sha69]. It is known to be one of the best — and asymptotically optimal — algorithms for solving the DLP in generic groups (groups without particular structure). And it was first discovered for the computation of class groups! Indeed, Shanks first described it in the context of class groups, for imaginary quadratic number fields. Factorization algorithms can also be derived from computing class groups — or at least a subgroup — in quadratic number fields (see [Coh93, Sections 8.6 & 10.2]).

Class groups also had a significant impact on one of the first partial proofs of Fermat's Last Theorem:

Theorem. *The equation* $x^n + y^n = z^n$ *has no non-trivial integer solutions for* $n \geq 3$.

Fermat's proof only covers the case $n = 4$ and suggests that one may instead address the equation $x^p + y^p = z^p$ for a prime number p and x, y, z pairwise relatively prime. The first proof in any generality came from Sophie Germain, who proved the case where x, y, and z are not multiples of p, for any prime p such that $2p+1$ is also prime: the so-called *Sophie Germain primes*. Before the complete proof provided by Wiles [Wil95] in 1994 — which uses elliptic curves — many proofs were proposed. Lamé in 1848 used the complex numbers satisfying $\zeta^p = 1$, now known as *roots of unity*, to decompose

$$x^p + y^p = (x+y)(x+\zeta y)(x+\zeta^2 y)\cdots(x+\zeta^{p-1} y). \tag{1}$$

However, his proof was not valid as Lamé assumed that the unique factorization property holds in the cyclotomic field $\mathbf{Q}(\zeta)$. Then, Kummer introduced (a notion close to) ideals to obtain the property of unique decomposition into prime ideals. This completed the proof for *regular primes*.

Definition. A prime number p is said to be *regular* if the order of the class group of $\mathbf{Q}(\zeta)$ is not a multiple of p, for $\zeta \neq 1$ such that $\zeta^p = 1$.

Then, combining Equation (1) with $x^p + y^p = z^p$ implies that every ideal generated by $(x+\zeta^k y)$, for $1 \leq k \leq p-1$, is a p-th power of an ideal $\mathfrak{a}_k$. For regular primes, as p does not divide the cardinality of the class group, then the principality of $\mathfrak{a}_k^p$ implies that $\mathfrak{a}_k$ is also principal. This trick is the key-point of the proof and the remaining part only consists in writing $x+\zeta y$ such that a contradiction arises.

It is still unknown whether or not there exist infinitely many regular primes. Jensen proved in 1915 that infinitely many primes are irregular. However, the regular primes are believed to have density $e^{-\frac{1}{2}}$ in the asymptotic sense of natural density, that is approximately 60 per cent (it is a conjecture of Siegel that has not been proven since 1964). For instance, Kummer noticed that the only irregular primes below 100 are 37, 59, and 67.

2 Contributions & Organization

The purpose of this thesis is to study and improve class group computations and to address related cryptographic questions. We present in Chapter 1 the mathematical tools we use in this thesis. We give the definitions of the class group and the units of a number field, in

order to be familiar with the structures we want to compute. A brief reminder on lattices and reduction is also included, as it is an essential part of our work. Chapter 2 is devoted to the history of class group computations, from Gauss to the state of the art. It is an opportunity to introduce methods that are now well known for problems that have been more thoroughly studied than class group computations. We describe the path from quadratic number fields to arbitrary-degree number fields over the years.

Our contributions to class group computations are presented in the three following chapters. In Chapter 3, we describe an algorithm to find a small defining polynomial of a number field. Indeed, there exist infinitely many polynomials defining a specific number field, but working with the one that has the smallest coefficients is usually better. We present this algorithm, along with its complexity analysis and study its impact in the context of class group computations. We notice that, considered as a precomputation, it extends a result of Biasse and Fieker [BF14] to wider classes of (small-degree) number fields.

In Chapter 4, we examine the general algorithm for class group computations. This one is used when we do not have a small defining polynomial for the number field. Based on a result of Biasse and Fieker [BF14], we simplify their algorithm, improve the complexity analysis and identify the optimal parameters to reduce the runtime. In Chapter 5, we focus on the conditional improvements made possible by a defining polynomial with small coefficients. In that case, we are able to present an algorithm whose runtime is considerably smaller than in the generic case. We also provide a result on finding a solution to the Principal Ideal Problem, that is recovering a generator of an ideal, assuming it is principal.

Finally, in Chapter 6, the cryptanalysis of a Fully Homomorphic Encryption scheme is presented. This attack corresponds to the solution of the Short Principal Ideal Problem in prime-power cyclotomic fields. This cryptosystem was described by Smart and Vercauteren at PKC 2010 [SV10] as a practical instantiation of the Gentry FHE scheme [Gen09]. We take advantage of the cyclotomic-field structure to reduce the dimension and use techniques described previously for recovering a generator. The final step that consists in deriving a short generator was already addressed by Cramer, Ducas, Peikert, and Regev at EUROCRYPT 2016 [CDPR16].

Publications

The results we have obtained led to the publication of articles in international conferences and journals. The thesis only tackles the first four of the following list, as they are the ones focusing on class group computations. They correspond respectively to Chapters 3, 4, 5, and 6. We briefly describe the other two in the next section.

[GJ16] **Reducing number field defining polynomials: an application to class group computations**
with Antoine Joux, *LMS Journal of Computation and Mathematics & ANTS XII, 2016.*

[GJ17a] **On the complexity of class group computations for large-degree number fields**
with Antoine Joux, *Submitted.*

[GJ17b] **Reducing the complexity for class group computations using small defining polynomials**
with Antoine Joux, *Submitted.*

[BEF+17] **Computing generator in cyclotomic integer rings — A subfield algorithm for the Principal Ideal Problem in L(1/2) and application to the cryptanalysis of a FHE scheme**
with Jean-François Biasse, Thomas Espitau, Pierre-Alain Fouque, and Paul Kirchner, *EUROCRYPT 2017.*

[GKL17] **Parametrizations for families of ECM-friendly curves**
with Thorsten Kleinjung and Arjen K. Lenstra, *ISSAC 2017.*

[GW17] **Loop-abort faults on supersingular isogeny cryptosystems**
with Benjamin Wesolowski, *PQCrypto 2017.*

Parallel contributions

ECM. The Elliptic Curve Method (ECM) for integer factorization was introduced in 1985 by H.W. Lenstra and published two years later in [Len87]. It is the asymptotically fastest method that has been published for finding relatively small factors of large composites. Although the number field sieve [LL93] is the most efficient general algorithm for integer factorization, there are two common use cases for ECM: it is widely used in attempts to find factors of large composites for which no information is available about the sizes of the prime factors (Propper found the largest ECM factor so far, a 274-bit factor of $7^{337}+1$) and it is used for the so-called *cofactoring* step of the number field sieve (where many relatively small composites have to be factored).

Given an odd composite integer N to be factored, ECM performs arithmetic operations on elliptic curves considered to be defined over the finite field $\mathbf{F}_p$ of cardinality p, for an unknown prime divisor p of N. It may find p if the cardinality of at least one of these curves over $\mathbf{F}_p$ is smooth. For this reason, we use curves that are known to have favorable smoothness properties, such as a large torsion group over $\mathbf{Q}$ or a cardinality that is divisible by a fixed factor. Constructions of ECM-friendly curves were published by Suyama [Suy85] (with a slight improvement by Montgomery in [Mon87, Section 10.3.2]), Atkin-Morain [AM93], and generalized by Bernstein *et al.* in [BBL10].

Originally formulated in [Len87] using Weierstrass curves, until around 2008 implementations of ECM mostly used Montgomery's approach from [Mon87]. With the introduction of Edwards curves [Edw07], a number of follow-up papers by Bernstein *et al.* [BBJ+08, BBLP13] ultimately led to "$a=-1$ *twisted Edwards*" curves by Hisil *et al.* [HWCD08] with torsion group isomorphic to $\mathbf{Z}/2\mathbf{Z}\times\mathbf{Z}/4\mathbf{Z}$ as one of the current most efficient ways to implement ECM, as shown by Bernstein *et al.* in [BBL10]. For these curves, Barbulescu *et al.* in [BBB+12] identify three families that all have the same larger smoothness probability and an even better fourth family. In [BBB+12] a parametrization is provided for one of the three equivalent families; the others are only illustrated by — a finite set of — small values found by enumeration. In particular a parametrization of the fourth and best family, which could lead to a better choice of curves for ECM, has so far not been published. By *parametrization* we mean that an elliptic curve along with a non-torsion point is determined as a function of some parameter: the parameter may be a point on some other elliptic curve or a rational number, thus giving rise to *elliptic* and *rational* parametrizations.

We extend the constructions from [BBB+12] by developing six further rational parametrizations, and use three of them to formulate five new elliptic parametrizations that enable fast generation of curves for all families of curves from [BBB+12]. We conduct the same tests as described in [BBL10] for the family of curves that are, based on their Galois properties, most promising for ECM. With respect to the criteria from [BBL10], usage of this family of curves leads to slightly better performance of ECM than reported before, with no significant fluctuations across curves from this same family. The newly parameterized curves may prove to be most useful for ECM-based cofactoring in the number field sieve.

SIDH. Cryptographic schemes based on supersingular elliptic curve isogenies were introduced in [CLG09] which proposed the hardness of path-finding in supersingular isogeny graphs and gave an application to cryptographic hash functions. It has since been used as an assumption for other cryptographic systems such as key-exchange and encryption [JF11]. Several other primitives have subsequently been built based on supersingular isogeny problems, such as zero-knowledge proofs of identity and signatures [FJP14, GPS16, JS14, STW12]. Efficient implementations of these primitives have rapidly followed: in software [CLN16, SIDH], in hardware [KAKJ17] and on embedded systems [KJA+16].

Even though the basic version of the key exchange protocol uses ephemeral secret values (as with classical Diffie-Hellman), some of these other schemes require static secret keys for at least one party. Such static secrets constitute primary material for active attacks, and such an attack was indeed described in [GPST16] that allows to find all n bits of the secret key with about n interactions with the victim. This attack can be prevented by the

Kirkwood *et al.* [KLM+15] validation method — which is essentially a Fujisaki–Okamoto transform [FO99] applied in the context of supersingular isogenies.

The results of [KJA+16], together with the fact that primitives based on supersingular isogenies enjoy significantly smaller keys than the other main candidates for post-quantum cryptography, suggest that they might be well-suited for use on embedded devices. This opens new avenues of potential side-channel attacks.

We describe the first side-channel attack against supersingular isogeny-based primitives, exploiting a fault-injection technique known as loop-abort fault injection, previously introduced for pairing based cryptography [PV06]. The iterative structure of isogeny computations render them susceptible to loop-abort fault attacks, allowing an attacker to recover all n bits of the key, within $O(n)$ interactions with the token and a negligible amount of computation. This attack is not prevented by any of the validation methods previously discussed for isogeny-based cryptosystems. Loop-abort fault injections were proven to be feasible in practice [BGG+14], and should therefore be taken into serious consideration when implementing schemes based on supersingular isogenies in a context susceptible to physical attacks.

Part I

Preliminaries

Chapter 1

Algebraic Number Theory Tools

Contents

The aim of this chapter is to familiarize the reader with the mathematical structures used in the rest of the thesis. We recall some basic results from number theory that are required for a proper understanding of the results presented in this thesis. This chapter is essentially derived from [Coh93, Section 4].

1.1 Linear algebra & Lattices

Before we start dealing with number theory, we provide a brief summary about Hermite and Smith Normal Forms. In addition, we give a reminder on lattices and reduction algorithms.

1.1.1 Hermite and Smith Normal Forms

Definition 1.1.1. An $m \times n$ matrix H with integer entries $h_{i,j}$ $(1 \leq i \leq m, 1 \leq j \leq n)$ is in *Hermite Normal Form* (HNF) if:

- H is upper triangular, *i.e.,* $h_{i,j} = 0$ for $i > j$, and the all-zero rows are located below any other row,
- The *pivot* — this is the first non-zero entry from the left — of a non-zero row is positive and always strictly to the right of the pivot of the row above it,
- The elements above pivots are non-negative and strictly smaller than the pivot.

Proposition 1.1.2. *For any $m \times n$ matrix M with coefficients in $\mathbf{Z}$, there exists a unique $m \times n$ matrix H in HNF of the form $H = UM$ with $U \in GL_n(\mathbf{Z})$ unimodular.*

Practically the computation of H is mainly performed modulo a large integer D: it is a manner to avoid an explosion of the size of the coefficients — see [Coh93, Section 2.4.2]. We only give one result about HNF computations because it is the one we use for our purposes. However, there exists a lot of results depending on the inputs and the requirements — fast or low-memory for instance.

In the following theorem, $\|M\| = \max|M_{i,j}|$ and ω denotes the matrix multiplication exponent. The smallest known value is $\omega = 2.3728639$ (see [Gal14]) but this result is only theoretical. A naive way leads to $\omega = 3$ while, in practice, we can make use of the Strassen algorithm [Str69] where $\omega = \log_2 7 \approx 2.807$. The function B is defined such that $B(t)$ bounds the number of bit operations to solve both the extended Euclidean problem with two t-bit integers and to apply the Chinese Remainder algorithm with moduli consisting of any two coprime integers less of at most t bits.

Theorem 1.1.3 ([SL96, Theorem 12])**.** *There exists a deterministic algorithm that takes as input an $m \times n$ rank-n integral matrix M, and produces as output the Hermite Normal Form H of M together with a unimodular pre-multiplier matrix U that satisfies $H = UA$. The runtime of the algorithm is bounded by*

$$O\left(n^{\omega-1} m \log\left(\frac{2m}{n}\right) B\left(\log\left(\frac{2m}{n}\right) n \log(n\|M\|)\right)\right) \text{ bit operations.}$$

In addition, we have a bound on the pre-multiplier as $\log\|U\| = O\left(\log\left(\frac{2m}{n}\right) n \log(n\|M\|)\right)$.

This algorithm also has good results in practice. The function B satisfies the inequality $B(t) \ll t(\log t)^2 \log\log t$, which adds a lot of logarithmic factors. Omitting them, we find a complexity that can be expressed as $\tilde{O}\left(n^\omega m \log\|M\|\right)$ in practice, for any practically applicable value of ω.

Besides being unique — and so being a representative of its equivalence class under unimodular multiplication — the HNF of an integral matrix provides a basis for its image. Its kernel is contained in the $m-n$ last rows of the pre-multiplier matrix U.

Definition 1.1.4. An $n \times n$ matrix S with integer entries $s_{i,j}$ is in *Smith Normal Form* (SNF) if S is a diagonal matrix with non-negative integer coefficients such that $s_{i,i} \mid s_{i+1,i+1}$ for all non-negative $i < n$.

Proposition 1.1.5. *For any $n \times n$ matrix M with coefficients in $\mathbf{Z}$ and rank n, there exists a unique $n \times n$ matrix S in SNF of the form $S = UMV$ with $U, V \in GL_n(\mathbf{Z})$ unimodular. We denote by d_i the i-th diagonal coefficient of S and the d_i are called* elementary divisors *of the matrix M.*

The Smith Normal Form is generally used for computing the structure of a finite abelian group. We know that every finite abelian group G is isomorphic to a direct sum of cyclic groups $\bigoplus \mathbf{Z}/d_i\mathbf{Z}$, with $d_i \mid d_{i+1}$. These d_i are exactly the elementary divisors greater than 1 of G, viewed as a $\mathbf{Z}$-module. This is how we are going to derive the structure of the class group.

Basically, an efficient way to compute the SNF is to begin by computing the HNF. Then, the cost comes down to the one of the HNF computation. Moreover, if we are only interested in the elementary divisors greater than 1, we can even only consider the $k \times k$ submatrix of the HNF where $h_{i,i} > 1$ for $i \leq k$ and $h_{i,i} = 1$ for $i > k$. This submatrix is called the *essential part* of H. In our context, this considerably reduces the size of the matrix involved.

Remark 1.1.6. The diagonal elements obtained after the HNF are usually *not* the elementary divisors.

Example 1.1.7.

$M = \begin{pmatrix} 2 & 1 \\ 0 & 2 \end{pmatrix}$ has elementary divisors 1 and 4, as $\begin{pmatrix} 1 & 0 \\ 0 & 4 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ -2 & 1 \end{pmatrix}\begin{pmatrix} 2 & 1 \\ 0 & 2 \end{pmatrix}\begin{pmatrix} 0 & -1 \\ 1 & 2 \end{pmatrix}$.

1.1.2 Lattices

A short definition we have for a lattice is the following one:

Definition 1.1.8. A *lattice* is a discrete additive subgroup of $\mathbf{R}^n$.

In addition to this group structure, we provide a norm on $\mathbf{R}^n$, usually the Euclidean norm. This induces the definition of an associated quadratic form on the lattice, namely $q : x \mapsto \sum x_i^2$, where the x_i $(1 \leq i \leq n)$ are the coordinates of x. Because a lattice is discrete, there necessarily exists a non-zero vector — not unique[1] — whose norm is minimal; this norm is called the *first minimum* of the lattice and is denoted $\lambda_1(\mathcal{L})$ for a lattice $\mathcal{L}$. This notion of shortness is the keystone of lattices.

A *basis* for a lattice $\mathcal{L}$ is a set of linearly independent vectors $b_1, \ldots, b_r$ belonging to $\mathcal{L}$ such that any vector in $\mathcal{L}$ can be written as a linear combination — with integer coefficients — of the vectors b_i. Any lattice $\mathcal{L}$ admits a basis, and the cardinality r is the same for all bases of a given lattice. It is called the *rank* of the lattice and is equal to the dimension of the vector subspace spanned by $\mathcal{L}$ in $\mathbf{R}^n$.

Given a basis B of a rank-r lattice $\mathcal{L}$ of $\mathbf{R}^n$, it can be represented by a $r \times n$ matrix where each row contains the coordinates of one vector in the basis. For two such bases B_1 and B_2, because they define the same lattice, there exists a unimodular matrix $U \in \mathrm{GL}_r(\mathbf{Z})$ such that $B_2 = UB_1$. As a consequence, the determinant $\det\left(B\,{}^tB\right)$ — where ${}^tB$ denotes the transpose matrix — is independent of the choice of B. Since it is a positive number, we obtain the following definition.

Definition 1.1.9. The *determinant* of the lattice $\mathcal{L}$ is defined as $\det \mathcal{L} = \sqrt{\det\left(B\,{}^tB\right)}$.

The matrix $B\,{}^tB$ itself is also important and is called the *Gram matrix* of the basis B. Indeed, it is equivalent to look at the lattice spanned by the basis B with the Euclidean norm and to look at the one defined by the trivial basis with the norm defined by the quadratic form $B\,{}^tB$.

1.1.3 Reduction algorithms

Lattice reduction is a research area that is still very active nowadays. It consists in finding a basis for a lattice $\mathcal{L}$ where the vectors are short and nearly orthogonal.

[1] If the vector $v \in \mathcal{L}$ is minimal, then $-v$, which also belongs to $\mathcal{L}$, is minimal as well.

The most common reduction algorithm is given in the work of Lenstra, Lenstra, and Lovász [LLL82]. Given as input any basis B_0 of a lattice and a parameter $\delta \in (0.25, 1)$, it outputs a δ-*LLL-reduced* basis $B = (b_1, \ldots, b_n)$, *i.e.,*

$$\forall i < j, \quad \left| < b_j | b_i^* > \right| \leq \frac{\|b_i^*\|^2}{2}, \tag{1.1}$$

$$\forall i, \quad \delta \|b_i^*\|^2 \leq \|b_{i+1}^*\|^2 + \frac{< b_{i+1} | b_i^* >^2}{\|b_i^*\|^2}, \tag{1.2}$$

where $< \cdot | \cdot >$ denotes the inner product and the vectors b_i^* result from the Gram-Schmidt orthogonalization of the basis B.

Let $\|B_0\|$ denote the bit-size of the input basis, that is the largest length of a basis vector under the Euclidean norm. The complexity of LLL-reduction for an integer lattice of rank r and dimension n is $O\left(nr^5(\log \|B_0\|)^3\right)$. It is a polynomial complexity so this algorithm is reasonably fast. However, using the special-case $\delta = \frac{3}{4}$, the bound we obtain for the first vector of the reduced basis is not so tight:

$$\|b_1\| \leq 2^{\frac{n-1}{2}} \lambda_1(\mathcal{L}). \tag{1.3}$$

Most of the time, this bound suffices and in practice, for small dimension ($n \leq 40$), the behavior of the LLL algorithm is much better than what Equation (1.3) makes us think.

If we really want to determine the first minimum and a shortest vector in the lattice, we need another method. This is the idea of enumeration algorithms, examined by Kannan in [Kan83]. Of course, we cannot perform a full enumeration of the vectors in the lattice: we begin by determining a set of candidates and enumerate them. This algorithm relies on the concept of *Hermite-Korkine-Zolotarev (HKZ) reduced basis*, which is a basis defined recursively by:

- the basis is *size-reduced* as in Equation (1.1),
- b_1 realizes the first minimum $\lambda_1(\mathcal{L})$,
- the projection of the vectors $b_2, \ldots, b_r$ orthogonally to b_1 form an HKZ reduced basis.

Hanrot and Stehlé [HS07, HS08] have improved this method to obtain a runtime in $\text{Poly}(\log \|B_0\|) \cdot r^{\frac{r}{2e}+o(r)}$. The gain on the quality of the output basis is then counterbalanced by the complexity becoming exponential in the rank r.

BKZ-reduction. Schnorr introduced in 1987 the *Block Korkine-Zolotarev (BKZ) reduction*. It is a balance between LLL and HKZ reductions. Roughly speaking, it consists in HKZ-reducing blocks that correspond to sublattices of fixed rank. Then the algorithm becomes polynomial

in the dimension, exponential in the block-size, and the shortness of the output-basis first vector depends on the block-size. The current best algorithms are the *slide-reduction* by Gama and Nguyen [GN08] and *Dual-BKZ* by Micciancio and Walter [MW16]. An accurate analysis of BKZ is also provided by Hanrot, Pujol, and Stehlé in [HPS11]. Finally, we obtain the following result:

Theorem 1.1.10. *The smallest vector v output by the BKZ algorithm with block-size β has a norm bounded by*

$$\|v\| \quad \leq \quad \beta^{\frac{n-1}{2(\beta-1)}} \cdot (\det \mathcal{L})^{\frac{1}{n}}.$$

The algorithm runs in time $\mathrm{Poly}(n, \log \|B_0\|) \left(\frac{3}{2}\right)^{\beta/2+o(\beta)}$*, where B_0 is the input basis.*

Proof. The bound we get is a direct consequence of [MW16, Theorem 1]. We only replaced the *Hermite constant* γ_β by an upper bound in $O(\beta)$. The cost analysis is derived from a quick study of [MW16, Algorithm 1], and the complexity of the *Shortest Vector Problem* (SVP) is below $\left(\frac{3}{2}\right)^{\beta/2+o(\beta)}$ operations, according to [BDGL16]. □

Cheon's trick. In a note [CL15] of 2015, Cheon and Lee suggest to convert the basis of an integer lattice having small determinant, to its HNF before reducing it. This method seems to be folklore, but this note gives a detailed analysis and we refer to it as *Cheon's trick*. We briefly develop here the idea and derive corresponding bounds.

Lemma 1.1.11. *Given $(b_1, \ldots, b_n)$ a basis in HNF of an n-dimensional lattice $\mathcal{L} \subset \mathbf{R}^n$, we have, for any $1 \leq i < n$,*

$$\det[b_1, \ldots, b_i] \leq \det[b_1, \ldots, b_{i+1}].$$

In particular, for any sublattice $\mathcal{L}'$ generated by the m first vectors $b_1, \ldots, b_m$, we have

$$\det \mathcal{L}' \leq \det \mathcal{L}.$$

We remark that both the n-th root of the determinant and an exponential factor in n appear in the bound of Theorem 1.1.10. In most cases, the term with the determinant prevails. However, when the determinant is small, the approximation factor can be larger. The idea behind Cheon's trick is then to reduce a lattice of smaller dimension in order to reduce this approximation factor. We fix the block-size $\beta \leq n$ and look at the output of BKZ performed on the sublattice $\mathcal{L}'$ generated by the m first vectors $b_1, \ldots, b_m$ of an HNF basis. From Lemma 1.1.11, we have

$$\|v\| \leq \beta^{\frac{m}{2\beta}} \cdot (\det \mathcal{L}')^{\frac{1}{m}} \leq \beta^{\frac{m}{2\beta}} \cdot (\det \mathcal{L})^{\frac{1}{m}}.$$

The condition we require on the determinant of the lattice is $\det \mathcal{L} \leq \beta^{\frac{n^2}{2\beta}}$: otherwise, for every $m \leq n$, the term $(\det \mathcal{L})^{\frac{1}{m}}$ is dominating. Assuming $\det \mathcal{L} \leq \beta^{\frac{n^2}{2\beta}}$, we identify the optimal sub-dimension m in $\{\beta, \ldots, n\}$ depending on β that minimizes this upper bound: it corresponds to the balance between the two factors, that is $m = \left\lfloor \sqrt{2\beta \log_\beta(\det \mathcal{L})} \right\rceil$. We fix m to this value and we obtain the following corollary.

Corollary 1.1.12. *For any integer lattice $\mathcal{L} \subset \mathbf{R}^n$ of rank n such that $\det \mathcal{L} \leq \beta^{\frac{n^2}{2\beta}}$, using BKZ reduction with block-size β along with Cheon's trick permits to output a short vector v that satisfies*

$$\log_\beta \|v\| \leq \sqrt{\frac{2}{\beta} \log_\beta(\det \mathcal{L})}\big(1 + o(1)\big).$$

This algorithm runs in time $\mathrm{Poly}(n, \log \|B_0\|) \cdot \left(\frac{3}{2}\right)^{\beta/2 + o(\beta)}$.

Proof. We consider the sublattice of dimension m, for m as defined above. The condition on the determinant of $\mathcal{L}$ ensures that our value of m is effectively lower than n. Then, by Theorem 1.1.10 and Lemma 1.1.11, we have

$$\|v\| \leq \beta^{\frac{m}{2\beta}} \cdot (\det \mathcal{L})^{\frac{1}{m}} = \beta^{\sqrt{(2/\beta)\log_\beta(\det \mathcal{L})}\left(1 + o(1)\right)},$$

which yields the announced result — the $(1 + o(1))$ factor appears because of the integer approximation of m. □

Remark 1.1.13. Thanks to Corollary 1.1.12, we want to point out that choosing block-size $\beta = \log(\det \mathcal{L})^{\frac{1}{3}}$ when it is smaller than n allows to describe an algorithm that runs in time $\mathrm{Poly}(n, \log \|B_0\|) \cdot \left(\frac{3}{2}\right)^{\beta/2 + o(\beta)}$ and outputs a vector of norm less than $\beta^{\sqrt{2}\beta\left(1 + o(1)\right)}$.

Reduction using the Gram matrix. In [EJ17], Espitau and Joux analyze approximate lattice reduction, using the Gram Matrix. Precisely, they emphasize that when we work with real lattices, it is better to make the approximations on the Gram matrix than on the Basis matrix. They provide an implementation that is able to ensure that the output is exact as long as the input precision is sufficient. It exits with error when the precision is insufficient.

1.2 Number fields & Polynomials

We begin number theory by giving the definition of a number field.

Definition 1.2.1. A *number field* $\mathbf{K}$ is a field containing $\mathbf{Q}$ which, considered as a $\mathbf{Q}$-vector space, is finite dimensional. The number $n = \dim_{\mathbf{Q}} \mathbf{K}$ is denoted by $[\mathbf{K} : \mathbf{Q}]$ and called the *extension degree* — or simply *degree* — of the number field $\mathbf{K}$.

1.2.1 Defining polynomials

The major theorem about number fields is the *Primitive Element Theorem*:

Theorem 1.2.2. *Let* $\mathbf{K}$ *be a number field of degree* n*. Then there exists a* $\theta \in \mathbf{K}$ *such that*

$$\mathbf{K} = \mathbf{Q}(\theta). \tag{1.4}$$

Such a θ *is called a* primitive element*. Its minimal polynomial is an irreducible polynomial of degree* n*.*

Definition 1.2.3. Let $\mathbf{K}$ be a number field of degree n. Every polynomial T that is the minimal polynomial of a primitive element $\theta \in \mathbf{K}$ is called a *defining polynomial* of $\mathbf{K}$. All defining polynomials are monic, irreducible and of degree n. They satisfy

$$\mathbf{K} \simeq \mathbf{Q}[X] / \langle T \rangle ,$$

that is the quotient ring of $\mathbf{Q}[X]$ modulo the ideal generated by the polynomial T.

Definition 1.2.4. All elements x in a number field $\mathbf{K}$ are *algebraic numbers*: there exists a polynomial P in $\mathbf{Q}[X]$ such that $P(x) = 0$, and P not identically zero. Among these elements, *algebraic integers* are the ones for which the polynomial P can be chosen monic and in $\mathbf{Z}[X]$. The set of all algebraic integers in $\mathbf{K}$ is a ring, called the *ring of integers* and denoted by $\mathcal{O}_{\mathbf{K}}$.

For our purposes, we only allow θ to be an algebraic integer in the notation of Equation (1.4). Hence, by defining polynomial, we refer to monic, irreducible, degree-n polynomials in $\mathbf{Z}[X]$.

1.2.2 Complex embeddings

The definition of number fields implies that they all may be embedded in $\mathbf{C}$. More precisely, we have the following result:

Proposition 1.2.5. *Let* $\mathbf{K}$ *be a number field of degree* n *and let* θ *be a primitive element. There exist exactly* n *field embeddings of* $\mathbf{K}$ *in* $\mathbf{C}$*, given by* $\theta \mapsto \theta_i$*, where the* θ_i *are the roots in* $\mathbf{C}$ *of the minimal polynomial of* θ*. These embeddings are* $\mathbf{Q}$*-linear, their images* $\mathbf{K}_i \subset \mathbf{C}$ *are called the* conjugate fields *of* $\mathbf{K}$ *and all the* $\mathbf{K}_i$ *are isomorphic to* $\mathbf{K}$*.*

Definition 1.2.6. The *signature* of a number field $\mathbf{K}$ is the pair (r_1, r_2) where r_1 is the number of embeddings of $\mathbf{K}$ whose image lies in $\mathbf{R}$ and $2r_2$ is the number of non-real complex embeddings, so that $r_1 + 2r_2 = n$. Note that the non-real embeddings always come in pairs since if σ_i is such an embedding, so is $\overline{\sigma_i}$.

We almost always order them in the following way: $\sigma_1, \ldots, \sigma_{r_1}$ for the real embeddings and $\sigma_{r_1+r_2+i} = \overline{\sigma_{r_1+i}}$ for $1 \leq i \leq r_2$. Finally, we get an embedding σ, called the *canonical embedding*,

$$\sigma : \mathbf{K} \longrightarrow \mathbf{R}^{r_1} \times \mathbf{C}^{r_2}.$$

For practical purpose, it is often considered as an $r_1 + 2r_2 = n$-tuple of real numbers.

Remark 1.2.7. When we want to work with a real lattice in $\mathbf{R}^n$, it suffices to differentiate the real and imaginary parts of the complex embeddings. However, we often add a factor $\sqrt{2}$ for these coordinates. Thus, as

$$|\sigma_i(x)|^2 + |\overline{\sigma_i}(x)|^2 = \left(\sqrt{2}\Re(\sigma_i(x))\right)^2 + \left(\sqrt{2}\Im(\sigma_i(x))\right)^2,$$

the L_2-norm is preserved through this operation from $\mathbf{C}^{r_2}$ to $\mathbf{R}^{2r_2}$. This map is often called the *Minkowski map*.

There exist efficient methods for determining the signature of a number field. It can e.g. be achieved using a result of Sturm by applying Euclid's algorithm to T and T' for a defining polynomial T (see [Coh93, Algorithm 4.1.11]). As we are more interested in the roots than in the signature itself, a simpler idea is to get an approximation of the roots and to count the number of real roots. However, Sturm's method makes it possible to compute the number of real roots without any approximation.

Example 1.2.8. The signature of a quadratic field is either $(2,0)$ in which case we speak of *real* quadratic fields, or $(0,1)$ in which case we speak of *imaginary* quadratic fields.

1.2.3 Representations of algebraic numbers

Now we have defined algebraic numbers and integers, we need to study the way for representing them. There exist different possibilities and we briefly describe the one that we are going to use.

Using their minimal polynomial. We begin by the tool we have already seen: the minimal polynomial. Obviously, every algebraic number x possesses a unique minimal polynomial T. This one is shared by all its *conjugates*, the n complex roots of T. We then need additional information to determine which of these roots is supposed to be represented. This can be accomplished using the numerical value of x as an element of $\mathbf{C}$, or at least an approximation; it suffices for this approximation to be closer to x than to any of its conjugates.

Using the vector-space structure. As $\mathbf{K}$ is a $\mathbf{Q}$-vector space, there exists a $\mathbf{Q}$-basis of $\mathbf{K}$. Let us denote by $(\theta_1, \ldots, \theta_n)$ such a basis. Then, every element can be written in the following way:

$$x = \frac{1}{d} \sum_{i=0}^{n-1} x_i \theta_i,$$

with $d \in \mathbf{N}^*$, $x_i \in \mathbf{Z}$ and $\gcd(x_0, \ldots, x_{n-1}, d) = 1$. In the case where $\theta_i = \theta^{i-1}$ for some primitive element θ, it is called the *standard representation* and it is the one we use most of the time. This choice also allows to perform efficiently operations between algebraic numbers.

Using matrices. Let $(\theta_1, \ldots, \theta_n)$ be a $\mathbf{Q}$-basis of $\mathbf{K}$. For every algebraic number x, the multiplication by x is an endomorphism of the vector space $\mathbf{K}$. Therefore x can be represented by the $n \times n$ matrix M_x of this endomorphism in that basis. Its coefficients are rationals but as in the previous case, we can identify a denominator and an integral matrix where all the integers involved are coprime. This choice induces longer runtime than standard representation for additions for instance, but is more suited for division.

Using the conjugates. Another method for representing an algebraic number is to use numerical approximations of its conjugates. Let σ_i denote the n distinct embeddings of $\mathbf{K}$ in $\mathbf{C}$. If $x = \sum_{i=0}^{n-1} x_i \theta^i$, with the $x_i \in \mathbf{Z}$, then

$$\sigma_j(x) = \sum_{i=0}^{n-1} x_i \sigma_j(\theta)^i,$$

and the $\sigma_j(x)$ are the conjugates of x, but in the specific order (the one chosen for ordering the embeddings). Hence every element can be represented by the $(r_1 + r_2)$-tuple

$$\sigma(x) = \left(\sigma_1(x), \ldots, \sigma_{r_1+r_2}(x)\right).$$

For practical purpose, $\sigma(x)$ is often considered as an $r_1 + 2r_2 = n$-tuple of real numbers.

In this representation, all operations are easy to perform, because they are done componentwise. However, we can work only with approximations and take care of round-off errors. To go back to an exact representation, Cohen explains a method in [Coh93, Section 4.2.4] that recovers the integers x_i from a good enough approximation.

1.2.4 Integral bases

Among all the $\mathbf{Q}$-basis of $\mathbf{K}$, some have additional properties. More precisely, we know that $\mathcal{O}_\mathbf{K}$ is a free $\mathbf{Z}$-module of rank n — which is only an abelian group with a basis — and for any

primitive element θ in $\mathbf{K}$, we have $\mathbf{Z}[\theta] \subset \mathcal{O}_{\mathbf{K}}$. However, this inclusion can be strict.

Definition 1.2.9. A $\mathbf{Z}$-basis of the free module $\mathcal{O}_{\mathbf{K}}$ is called an *integral basis* of $\mathbf{K}$.

Integral bases play an important role because in number fields, we are most often concerned by algebraic integers rather than algebraic numbers. Therefore, we are always trying to work with such a basis. There exist practical algorithms for computing an integral basis. It is completely out of our topic as we always consider that we have such a basis as inputs. We only give a sketchy idea of the way to do it, details are available in [Coh93, Section 6.1]. The method consists in enlarging the module $\mathbf{Z}[\theta]$ prime by prime until we have reached the full ring of integers $\mathcal{O}_{\mathbf{K}}$ — for θ a primitive element. More details are given in Remark 1.2.14.

1.2.5 Discriminants

In the context of number fields, two important notions of discriminant appear. The discriminant of the defining polynomial T denoted by $\Delta(T)$ and the discriminant of the number field $\mathbf{K}$ denoted by $\Delta_{\mathbf{K}}$. These values are related but different in general.

Definition 1.2.10. Given any integral basis $\omega_1, \ldots, \omega_n$ of $\mathcal{O}_{\mathbf{K}}$, the *discriminant* of $\mathbf{K}$ is the square of the determinant of the $n \times n$ matrix B whose entries are $B_{i,j} = \sigma_i(\omega_j)$, where the σ_i are the n complex embeddings. This can be written as

$$\Delta_{\mathbf{K}} = \det\left(\sigma_i(\omega_j)\right)^2.$$

Note that this value is independent of the choice of the integral basis and corresponds to the determinant of the canonical embedding of $\mathcal{O}_{\mathbf{K}}$.

Special case of dimension 2. For quadratic fields, the discriminant carries all the information. To distinguish quadratic fields from higher-degree number fields, we use D to denote the discriminant of quadratic fields.

Definition 1.2.11. An integer D is called a *fundamental discriminant* if one of the following statements holds:

- $D \equiv 1 \bmod 4$ and D is square-free,
- $D = 4m$, where $m \equiv 2, 3 \bmod 4$ and m is square-free.

Proposition 1.2.12. *Let D be a fundamental discriminant. Then D is the discriminant of the quadratic field $\mathbf{K} = \mathbf{Q}(\sqrt{D})$ and an integral basis of $\mathbf{K}$ is given by $\left(1, \frac{D+\sqrt{D}}{2}\right)$.*

For the purpose of simplification, we fix our monic irreducible degree-n polynomial T as

$$T(X) = \sum_{k=0}^{n} t_k X^k = t_n \prod_{j=1}^{n} (X - \tau_j) \qquad \text{with } t_n = 1. \tag{1.5}$$

Definition 1.2.13. The discriminant of T is defined as

$$\Delta(T) = (-1)^{\frac{n(n-1)}{2}} \frac{1}{t_n} \mathrm{Res}(T, T'),$$

where $\mathrm{Res}(T, T')$ is the resultant of T and its derivative. Note that, despite the fact that we only consider monic polynomials T, we give here the general formula that includes the leading term t_n.

The link with the discriminant of the field $\mathbf{K}$ comes from the fact that the discriminant of a monic polynomial T corresponds to the discriminant of the suborder[2] of $\mathcal{O}_\mathbf{K}$ defined by T, *i.e.,* $\mathbf{Z}[\theta]$ where θ is a root of T. This implies

$$\Delta(T) = C^2 \Delta_\mathbf{K},$$

where $C \in \mathbf{N} \setminus \{0\}$ is the *index* of the suborder $C = \left[\mathcal{O}_\mathbf{K} : \mathbf{Z}[\theta]\right]$.

Remark 1.2.14. This index C plays a key role in the construction of the integral basis. Indeed, when we enlarge $\mathbf{Z}[\theta]$ primes by primes to reach $\mathcal{O}_\mathbf{K}$, the involved primes are factors of C. Therefore, given $\Delta(T)$, the candidates are the primes whose square divide it. Hence, the costly part of the algorithm is the factorization of the discriminant of the input polynomial.

Our first result is a way to bound the discriminant of a number field by something depending on the size of the coefficients of a defining polynomial. To do that, we introduce two quantities, defined from the polynomial.

Definition 1.2.15. For a polynomial T as in Equation (1.5), we define

- the *height* of T as $H(T) = \max_k |t_k|$,
- the *Mahler measure* of T as

$$M(T) = |t_n| \prod_{j=1}^{n} \max\left(1, |\tau_j|\right).$$

The *Mahler measure* was introduced by Mahler in [Mah60], but before him, something similar appeared in the work of Lehmer [Leh33]. He defined it as $\log M(T) = \int_0^1 \log|T(e^{2i\pi t})|\mathrm{dt}$,

[2] Here, the fact that T is monic is essential. Otherwise, $\mathbf{Z}[\theta]$ would not be a suborder of $\mathcal{O}_\mathbf{K}$.

but we prefer to work with this alternative form, directly obtained thanks to Jensen's formula. Again, despite considering monic polynomials, we include t_n for the sake of generality.

The link between the Mahler measure and the height of a polynomial is quite tight and can be illustrated by the two following inequalities. First, Mahler shows in [Mah60] that for all $k \in \{0,\dots,n\}$, we have $|t_k| \leq \binom{n}{k} M(T)$ thus

$$H(T) \leq \binom{n}{\lfloor \frac{n}{2} \rfloor} \cdot M(T) \leq 2^n \cdot M(T) \tag{1.6}$$

In addition, it is proven in [MG94] that

$$M(T) \leq \left(\sum_{i=0}^{n} |t_i|^2 \right)^{\frac{1}{2}} \quad \text{which implies} \quad M(T) \leq \sqrt{n+1}\, H(T). \tag{1.7}$$

We desire to find an inequality between $\Delta_{\mathbf{K}}$ and $H(T)$ for a defining polynomial T of $\mathbf{K}$. We begin by looking for a relation between the discriminant $\Delta(T)$ and the height $H(T)$. The resultant of T and T' can be computed as the determinant of the $(2n-1) \times (2n-1)$ Sylvester matrix whose entries are all bounded by $nH(T)$ in absolute value. It then follows from Hadamard's inequality

$$|\Delta_{\mathbf{K}}| \leq |\Delta(T)| \leq \left(n\sqrt{2n-1}\, H(T) \right)^{2n-1}. \tag{1.8}$$

Using the Mahler measure, it is possible to refine the bound on $|\Delta(T)|$ we have derived from Hadamard's inequality (see [Mah64, Theorem 1]) and to obtain

$$|\Delta(T)| \leq n^n M(T)^{2n-2}. \tag{1.9}$$

Proposition 1.2.16. *Let T be a monic irreducible polynomial of degree $n \geq 2$. Then T defines a number field $\mathbf{K}$ whose discriminant $\Delta_{\mathbf{K}}$ satisfies*

$$|\Delta_{\mathbf{K}}| \leq |\Delta(T)| \leq n^{2n} H(T)^{2n-2}.$$

Proof. It is an almost direct consequence of combining (1.7) and (1.9) which yields the improved bound for the discriminant. It only remains to check the simple fact that

$$\forall n \in \mathbf{N}^*, \quad (n+1)^{n-1} \leq n^n.$$

□

1.3 Orders & Ideals

Before introducing the class group, the central point of this thesis, we need to define the main building blocks of it, *ideals*.

Definition 1.3.1. An *order* $\mathcal{O}$ in $\mathbf{K}$ is a subring of $\mathbf{K}$ which, as a $\mathbf{Z}$-module, is finitely generated and of maximal rank $n = \deg \mathbf{K}$.

By definition, all orders $\mathcal{O}$ are subrings of $\mathcal{O}_\mathbf{K}$. The ring of integers $\mathcal{O}_\mathbf{K}$ is then the *maximal order* of $\mathbf{K}$ and this terminology explains the notation. Henceforth we use both *ring of integers* or *maximal order* for $\mathcal{O}_\mathbf{K}$. The majority of the content of this thesis may be applied in a suborder of $\mathcal{O}_\mathbf{K}$. However, because it is always nicer to be in $\mathcal{O}_\mathbf{K}$, we are only concerned by this choice.

Example 1.3.2. In a quadratic field $\mathbf{Q}(\sqrt{D})$, all the orders are uniquely determined by their discriminant Df^2, for an integer $f \geq 1$, called the *conductor*.

Definition 1.3.3.

- An *ideal* $\mathfrak{a}$ of $\mathcal{O}_\mathbf{K}$ is a sub-$\mathcal{O}_\mathbf{K}$-module of $\mathcal{O}_\mathbf{K}$, *i.e.,* a sub-$\mathbf{Z}$-module of $\mathcal{O}_\mathbf{K}$ such that for every $x \in \mathcal{O}_\mathbf{K}$ and $a \in \mathfrak{a}$ we have $xa \in \mathfrak{a}$.
- A *fractional ideal* $\mathfrak{a}$ in $\mathcal{O}_\mathbf{K}$ is a non-zero submodule of $\mathbf{K}$ such that there exists a non-zero integer d with $d\mathfrak{a}$ ideal of $\mathcal{O}_\mathbf{K}$. To make a clear distinction between ideals and fractional ideals, we occasionally refer to the former as *integral ideals*.
- An ideal (fractional or not) is said to be a *principal ideal* if there exists $x \in \mathbf{K}$ such that $\mathfrak{a} = x\mathcal{O}_\mathbf{K}$. For sake of simplicity, when the order $\mathcal{O}_\mathbf{K}$ is fixed, such an ideal generated by $x \in \mathbf{K}$ is denoted by $\langle x \rangle$.

We define the product of two ideals as

$$\mathfrak{a}\mathfrak{b} = \left\{ \sum_{i=1}^{k} a_i b_i \mid k \in \mathbf{N}, a_i \in \mathfrak{a}, b_i \in \mathfrak{b} \right\}.$$

Because we work in the maximal order $\mathcal{O}_\mathbf{K}$, every fractional ideal $\mathfrak{a}$ is invertible and

$$\mathfrak{a}^{-1} = \{x \in \mathbf{K} \mid x\mathfrak{a} \subset \mathcal{O}_\mathbf{K}\}.$$

Proposition 1.3.4. *Let $\mathcal{I}(\mathbf{K})$ be the set of fractional ideals of $\mathcal{O}_\mathbf{K}$. Then $\mathcal{I}(\mathbf{K})$ is an abelian group.*

As in the case of $\mathbf{Z}$, there exist *prime* elements in $\mathcal{O}_\mathbf{K}$.

Definition 1.3.5. An ideal $\mathfrak{p}$ of $\mathcal{O}_\mathbf{K}$ is called a *prime ideal* if $\mathfrak{p} \neq \mathcal{O}_\mathbf{K}$ and if the quotient ring $\mathcal{O}_\mathbf{K}/\mathfrak{p}$ is an integral domain.

Example 1.3.6. In $\mathbf{Z}$, the prime ideals are exactly the $p\mathbf{Z}$ for p prime and $\{0\}$.

The existence of these prime ideals allows us to define a *prime decomposition* property as the one that holds in $\mathbf{Z}$:

Proposition 1.3.7. *Every fractional ideal $\mathfrak{a}$ can be written in a unique way as*

$$\mathfrak{a} = \prod_{\mathfrak{p}} \mathfrak{p}^{v_\mathfrak{p}(\mathfrak{a})},$$

the product being over a finite set of prime ideals and the exponents $v_\mathfrak{p}(\mathfrak{a})$, called the $\mathfrak{p}$-adic valuation*, being in $\mathbf{Z}$. In particular, $\mathfrak{a}$ is an integral ideal if and only if all the valuations $v_\mathfrak{p}(\mathfrak{a})$ are non-negative.*

We now provide a small dictionary that shows that number fields are only a generalization of what happens in $\mathbf{Q}$. These result may be easily derived from the $\mathfrak{p}$-adic valuation expressions.

fractional ideal	$\longleftrightarrow$	rational number
integral ideal	$\longleftrightarrow$	integer
inclusion	$\longleftrightarrow$	divisibility (reverse order)
sum	$\longleftrightarrow$	greatest common divisor
intersection	$\longleftrightarrow$	lowest common multiple
product	$\longleftrightarrow$	product

We define *ideal divisibility* as for integers in order to introduce prime decomposition, even though it exactly corresponds to inclusion. In dimension 2, we have a little more structure on the ideals thanks to quadratic forms. This is develop in Section 2.1.1.

Coefficient embedding and ideal lattices. We have seen with the ring of integers that there exists a canonical embedding $\sigma : \mathcal{O}_\mathbf{K} \to \mathbf{R}^{r_1} \times \mathbf{C}^{r_2}$. It also applies to every ideal of $\mathcal{O}_\mathbf{K}$. In addition, there exists another embedding ς, called the *coefficient embedding*. Let $\omega_1, \ldots, \omega_n$ be a fixed integral basis of $\mathcal{O}_\mathbf{K}$ and let $\mathfrak{a}$ be an ideal of $\mathcal{O}_\mathbf{K}$. Then, for every $x \in \mathfrak{a}$, we have $x = \sum x_i \omega_i$ so that we define $\varsigma(x) = (x_1, \ldots, x_n) \in \mathbf{Z}^n$. As every ideal $\mathfrak{a} \subset \mathcal{O}_\mathbf{K}$ is generated as a $\mathbf{Z}$-module by n elements, we obtain an $n \times n$ matrix by considering the coefficient embedding of each element of the generating family. This matrix defines a lattice, called the *ideal lattice* associated to $\mathfrak{a}$.

Example 1.3.8. For $\mathcal{O}_\mathbf{K}$ viewed as an ideal, we obtain the trivial lattice. Therefore we only talk about coefficient embedding for ideals, while the canonical embedding over $\mathcal{O}_\mathbf{K}$ brings the definition of the discriminant.

Remark 1.3.9. The canonical embedding of an ideal $\mathfrak{a}$ can be directly derived from the product between its coefficient embedding and the canonical embedding of the ring of integers $\mathcal{O}_\mathbf{K}$.

Representations of ideals. From the coefficient embedding defined above, we may derive easily one of the ways to represent ideals. More precisely, we require an additional formatting because we consider the HNF of the ideal lattices. With this representation, it is possible to compute the sums and products between ideals, looking respectively at the HNF of $n \times 2n$ or $n \times n^2$ matrices. Note that such a representation depends on the choice made for the integral basis.

Using the stronger $\mathcal{O}_\mathbf{K}$-module structure allows us to infer another way to carry the information.

Proposition 1.3.10. *Let $\mathfrak{a}$ be an integral ideal of $\mathcal{O}_\mathbf{K}$. There exists a non-zero element in $\mathfrak{a} \cap \mathbf{Z}$. Denoting by $\ell(\mathfrak{a})$ the smallest positive element of $\mathfrak{a} \cap \mathbf{Z}$, there exists an element β such that*

$$\mathfrak{a} = \ell(\mathfrak{a})\mathcal{O}_\mathbf{K} + \beta\mathcal{O}_\mathbf{K}.$$

Here, we represent an ideal by giving two generators. It is called a *two element representation* and is denoted by $\langle \ell(\mathfrak{a}), \beta \rangle$. We force here the first element to be as simple as possible but for every non-zero $\alpha \in \mathfrak{a}$, there exists a $\beta \in \mathfrak{a}$ such that $\mathfrak{a} = \langle \alpha, \beta \rangle$. Although this representation requires much less storage, it is not convenient for classical operations, except for two particular cases: in quadratic fields ($n = 2$) or for prime ideals.

Prime ideals. This is a brief overview of the results we need about prime ideals. Let $\mathbf{K}$ be a number field of degree n.

Proposition 1.3.11. *If $\mathfrak{p}$ is a prime ideal of $\mathbf{K}$, then $\mathfrak{p} \cap \mathbf{Z} = p\mathbf{Z}$ for some prime number p. We say that $\mathfrak{p}$ is a prime ideal* above *p and that p is* below *$\mathfrak{p}$.*

Theorem 1.3.12. *Let p be a prime number. There exist positive integers e_i such that*

$$p\mathcal{O}_\mathbf{K} = \prod_{i=1}^{g} \mathfrak{p}_i^{e_i},$$

where the $\mathfrak{p}_i$ are all the prime ideals above p.

The integer e_i is called the *ramification index* of p at $\mathfrak{p}_i$. The degree of the field extension $f_i = \left[\mathcal{O}_\mathbf{K}/\mathfrak{p}_i : \mathbf{Z}/p\mathbf{Z}\right]$ is called the *residual degree* — or simply the *degree* — of $\mathfrak{p}_i$.

Proposition 1.3.13. *With the notation as above, we have the following equality*

$$\sum_{i=1}^{g} e_i f_i = n.$$

Combining these specific results with the Proposition 1.3.10, we can obtain a more precise representation for prime ideals:

Theorem 1.3.14. *Let* $\mathbf{K}$ *be a number field,* T *a defining polynomial and* θ *a primitive element associated to* T*. Then for any prime* p *not dividing the index* $f = \left[\mathcal{O}_\mathbf{K} : \mathbf{Z}[\theta]\right]$*, we obtain a prime decomposition for* $p\mathcal{O}_\mathbf{K}$ *as follows. Let*

$$T = \prod_{i=1}^{g} T_i^{e_i} \mod p$$

be the decomposition of T *into monic irreducible factors over* $\mathbf{F}_p$*. Then we have*

$$p\mathcal{O}_\mathbf{K} = \prod_{i=1}^{g} \mathfrak{p}_i^{e_i},$$

where $\mathfrak{p}_i = \left\langle p, T_i(\theta) \right\rangle = p\mathcal{O}_\mathbf{K} + T_i(\theta)\mathcal{O}_\mathbf{K}$.
Furthermore, the residual degree f_i *of* $\mathfrak{p}_i$ *is equal to the degree of* T_i*.*

We have a similar result for the cases where p divides f but it requires more material — the same as for integral bases (see [Coh93, Section 6.2]). However, practically, the index is very small — often it is equal to 1 — and divisible by only a very tiny set of primes, so Theorem 1.3.14 is an important result for the representation of prime ideals.

1.4 Norms & Smoothness

1.4.1 Norms in K

Let x be an algebraic number. Its *norm* is by definition the product of its conjugates. If $T = \sum_{i=0}^{m} t_i X^i$ is the minimal polynomial of x, for a divisor m of n, then we have

$$\mathcal{N}(x) = (-1)^m \frac{t_0}{t_m}. \tag{1.10}$$

Usually, we consider the norms with respect to a fixed number field. If $\mathbf{K} = \mathbf{Q}(x)$ the definition of the norm suits Equation (1.10) but if $\mathbf{Q}(x) \subsetneq \mathbf{K}$ care must be taken for keeping $\mathcal{N}$ multiplicative.

Proposition 1.4.1. *Let K be a number field of degree n and σ_i the n distinct embeddings of $\mathbf{K}$ in $\mathbf{C}$. If $x \in \mathbf{K}$ has degree m, we have*

$$\mathcal{N}_{\mathbf{K}/\mathbf{Q}}(x) = \prod_{1 \le i \le n} \sigma_i(x) = \mathcal{N}(x)^{n/m}$$

and for any x and y in $\mathbf{K}$,

$$\mathcal{N}_{\mathbf{K}/\mathbf{Q}}(x) \cdot \mathcal{N}_{\mathbf{K}/\mathbf{Q}}(y) = \mathcal{N}_{\mathbf{K}/\mathbf{Q}}(x \cdot y).$$

We easily derive a first relation between the norm of an algebraic integer and the norm of its canonical embedding, based on the inequality of arithmetic and geometric means.

Lemma 1.4.2. *For every algebraic integer $x \in \mathcal{O}_{\mathbf{K}}$, we have*

$$\mathcal{N}_{\mathbf{K}/\mathbf{Q}}(x) \le \left(\frac{\|\sigma(x)\|}{\sqrt{n}} \right)^n.$$

This definition of the norm is very convenient for algebraic numbers represented as the vector of their conjugates. We also give expressions of the norm suitable for other representations.

Proposition 1.4.3.

- *For an algebraic number given by its standard representation $x = \frac{1}{d} \sum_{i=0}^{n-1} x_i \theta^i$, where θ is a primitive element, the norm of x satisfies*

$$\mathcal{N}_{\mathbf{K}/\mathbf{Q}}(x) = d^{-n} \operatorname{Res}(T, P_x), \tag{1.11}$$

  where $\operatorname{Res}(T, P_x)$ denotes the resultant of $P_x = \sum_{i=0}^{n-1} x_i X^i$ and T, the defining polynomial associated to θ.

- *For an algebraic number x represented by the multiplication-by-x matrix M_x, the norm is given by*

$$\mathcal{N}(x) = \pm \det M_x.$$

The bounds for the resultants displayed in [BL10, Theorem 7] allow us to provide another bound on the field norm of an element given in standard representation:

Lemma 1.4.4. *For an algebraic integer $x = P_x(\theta)$ for $P_x \in \mathbf{Z}[X]$ and θ a root of the defining polynomial T of $\mathbf{K}$, we know that*

$$|\mathcal{N}_{\mathbf{K}/\mathbf{Q}}(x)| \le (n+1)^{m/2} (m+1)^{n/2} H(P_x)^n H(T)^m,$$

where $n = \deg T = \deg \mathbf{K}$ and $m = \deg P_x = \deg x$.

1.4.2 Ideal norms

One can generalize the notion of norm of an element in the number field to the norm of an integral ideal.

Proposition 1.4.5. *Let $\mathfrak{a}$ be a non-zero ideal of $\mathcal{O}_\mathbf{K}$. The quotient $\mathcal{O}_\mathbf{K}/\mathfrak{a}$ is a finite ring and its cardinality is called the* norm *of $\mathfrak{a}$ and denoted $\mathcal{N}(\mathfrak{a})$.*

Because we work in the maximal order $\mathcal{O}_\mathbf{K}$, all ideals are invertible so that the ideal norm is multiplicative, but this is not true in all orders.

Proposition 1.4.6. *For $\mathfrak{a}$ and $\mathfrak{b}$ two ideals of $\mathcal{O}_\mathbf{K}$, we have*

$$\mathcal{N}(\mathfrak{a}\cdot\mathfrak{b}) = \mathcal{N}(\mathfrak{a})\cdot\mathcal{N}(\mathfrak{b}).$$

Moreover this norm is closely linked to the norm of integers in the sense that for every $x \in \mathcal{O}_\mathbf{K}$,

$$\mathcal{N}\big(\langle x\rangle\big) = \big|\mathcal{N}_{\mathbf{K}/\mathbf{Q}}(x)\big|. \tag{1.12}$$

Note that one can dispense with the absolute value in the later equality by using *Archimedian valuations* — also called *infinite places* — but it is not necessary for our purposes.

As a result, we can directly relate the norm of the embedding to the field norm using Lemma 1.4.4:

Corollary 1.4.7. *For any $x \in \mathcal{O}_\mathbf{K}$, using the coefficient embedding ς, we have the inequality $\big|\mathcal{N}_{\mathbf{K}/\mathbf{Q}}(x)\big|^{1/n} \leq (n+1)\cdot H(T)\cdot\|\varsigma(x)\|$,*

Another important result for ideal norms is the following one concerning prime ideals.

Proposition 1.4.8. *Let $\mathfrak{p}$ be a prime ideal of $\mathcal{O}_\mathbf{K}$ above p whose residual degree is f. The norm of $\mathfrak{p}$ satisfies*

$$\mathcal{N}(\mathfrak{p}) = p^f.$$

The norm of an ideal $\mathfrak{a}$ can be used to give an upper bound on the norm of the smallest non-zero element it contains. There always exists a non-zero $x \in \mathfrak{a}$ for which

$$\big|\mathcal{N}_{\mathbf{K}/\mathbf{Q}}(x)\big| \leq \left(\frac{2}{\pi}\right)^{r_2}\sqrt{|\Delta_\mathbf{K}|}\,\mathcal{N}(\mathfrak{a}),$$

where $\Delta_\mathbf{K}$ is the discriminant of $\mathbf{K}$ and r_2 is the number of pairs of complex embeddings, defined as previously.

Moreover, ideal norm acts as a proportionality coefficient between the norm of an ideal and the determinant of its embedding.

Lemma 1.4.9. *For any integral ideal* $\mathfrak{a}$ *of* $\mathbf{K}$, $\sigma(\mathfrak{a})$ *is a lattice of* $\mathbf{R}^n$ *and*

$$\det \sigma(\mathfrak{a}) = \sqrt{|\Delta_{\mathbf{K}}|}\,\mathcal{N}(\mathfrak{a}).$$

1.4.3 Smooth integers

The most common notion of size of an integer n is its binary length, which is the quantity $\lceil \log_2 n \rceil$. There are however other useful measures, and one of particular interest in the present context is the sizes of the primes dividing n compared to n.

Definition 1.4.10. For an integer $B \in \mathbf{N}$, we say that an integer is B-smooth if all its prime factors are below B. The bound B is then often called a *smoothness bound*.

Before the results about smoothness probability, we give an estimation of the size of the primes involved in the prime decomposition of an integer.

Proposition 1.4.11. *Let* n *be an integer and* $p_r \leq \cdots \leq p_1$ *primes such that* $n = \prod_{1 \leq i \leq r} p_i$. *We have an estimation of the* asymptotic average relative size *of the largest primes involved in the decomposition:*

$$\log_n p_1 = 0.6243299885 \tag{1.13}$$

$$\log_n p_2 = 0.2095808743 \tag{1.14}$$

$$\log_n p_3 = 0.0883160989. \tag{1.15}$$

Proof. This result is provided in [KP76, Section 9]. These values are derived from the probability that the factor p_k of $n \in \{1, \cdots, N\}$ satisfies $p_k < n^x$, for x in $[0,1]$, when N tends to infinity. □

Smoothness probability. Let us denote by $\mathcal{P}(x,y)$ the probability that an integer x is y-smooth, that means all prime factors of x are less than or equal to y. Dickman was the first one to address the question of asymptotic formulae in [Dic30]. Before stating his result, we introduce the Dickman *rho*-function, defined over $\mathbf{R}^+$ as the unique continuous function that satisfies $u\rho'(u) + \rho(u-1) = 0$ with initial condition $\rho(u) = 1$ for $u \in [0,1]$.

Proposition 1.4.12. *For any fixed* $u > 0$, *we have*

$$\lim_{x \to \infty} \mathcal{P}\left(x, x^{1/u}\right) = \rho(u).$$

Proof. This result appears in the work of Dickman [Dic30] and in the survey written later by Hildebrand and Tenenbaum [HT93]. The latter also showed [HT93, Corollary 1.3] that when u is large enough, $\rho(u)$ may be approximated by $u^{-u(1+o(1))}$. □

The main drawback of that previous result is that u has to be fixed: it cannot depend on x. This issue is covered by the stronger result of Canfield, Erdős, and Pomerance in [CEP83]:

Theorem 1.4.13. *For every $\varepsilon > 0$, there exists a constant C_ε such that for all $x \geq 1$ and u satisfying $3 \leq u \leq (1-\varepsilon)\frac{\log x}{\log\log x}$, we have*

$$\mathcal{P}(x, x^{1/u}) \geq e^{-u\left(\log u + \log\log u - 1 + \frac{\log\log u - 1}{\log u} + E(x,u)\right)},$$

where

$$|E(x,u)| \leq C_\varepsilon \left(\frac{\log\log u}{\log u}\right)^2.$$

Eventually, we can express $\mathcal{P}(x,y)$ by fixing u such that $u = \frac{\log x}{\log y}$ and substitute in the last expression. We obtain

$$\mathcal{P}(x,y) = u^{-u(1+o(1))},$$

which we already have from Dickman's work.

Subexponential L-notation. The term *subexponential* has been introduced for describing algorithms whose complexity is larger than polynomial but smaller than exponential. More precisely, the amount of time required for completing such an algorithm for an n-bit input is about $2^{O(n^\alpha)}$, for $0 < \alpha < 1$. Thus, we introduce the handy L-notation used for expressing subexponential complexities.

Definition 1.4.14. Given two constants α and c with $\alpha \in [0,1]$ and $c > 0$, $L_N(\alpha, c)$ is used as a shorthand for

$$e^{(c+o(1))(\log N)^\alpha (\log\log N)^{1-\alpha}},$$

where $o(1)$ tends to 0 as N tends to infinity. We sometimes encounter the notation $L_N(\alpha)$ when specifying c is superfluous, that is considering quantities in $L_N(\alpha, O(1))$.

We also introduce

$$Ł_N(\alpha, c) = e^{c(\log N)^\alpha (\log\log N)^{1-\alpha}},$$

without the $o(1)$ for considering constants — for algorithm inputs for instance.

Remark 1.4.15. The reason of the factor in $\log\log N$ is complexity calculation. Indeed, this is exactly what we want for describing the complexity of an algorithm based on index calculus method (see Section 2.3).

Before translating the Canfield-Erdős-Pomerance theorem in terms of L-notation, we give easy calculation rules for this notation.

Proposition 1.4.16. *Let $\alpha_1, \alpha_2 \in [0,1]$ and $c_1, c_2 > 0$. Then we have*

$$L_{L_N(\alpha_2,c_2)}(\alpha_1, c_1) = L_N\left(\alpha_1\alpha_2, c_1 c_2^{\alpha_1}\alpha_2^{1-\alpha_1}\right) \tag{1.16}$$

$$L_N(\alpha_1, c_1)\cdot L_N(\alpha_2, c_2) = \begin{cases} L_N(\alpha_1, c_1) & \text{if } \alpha_1 > \alpha_2 \\ L_N(\alpha_1, c_1 + c_2) & \text{if } \alpha_1 = \alpha_2 \end{cases} \tag{1.17}$$

Corollary 1.4.17. *Assuming that $x = Ł_N(\alpha_1, c_1)$, $y = Ł_N(\alpha_2, c_2)$, and $\alpha_1 > \alpha_2$, Theorem 1.4.13 can be expressed as*

$$\mathcal{P}(x,y) = L_N\left(\alpha_1 - \alpha_2, (\alpha_1 - \alpha_2)\frac{c_1}{c_2}\right)^{-1}.$$

Smoothness tests. Now we have estimated the ratio of smooth numbers below N to N, it remains to give a way to recognize them. We need to introduce *smoothness tests*. The first idea one may have is considering the complete factorization. Once we know the prime decomposition of an integer, it is easy to recognize if the number is smooth with respect to some smoothness bound. The best algorithm for factoring an integer N is currently the *Number Field Sieve* (NFS) and has runtime in $L_N\left(\frac{1}{3}, \sqrt[3]{\frac{64}{9}}\right)$ — see [LLMP90] for more details.

However it seems reasonable that, given a smoothness bound B, to test if an integer is B-smooth or not has a complexity that essentially depends on B, and not so much on the input integer. Such an algorithm exists and is derived from the *Elliptic Curve Method*, introduced by Lenstra in [Len87] for factoring integers. It provides a Monte-Carlo algorithm whose heuristic complexity is given in the following proposition.

Proposition 1.4.18. *For a given smoothness bound B and an integer N, ECM finds the B-smooth part of N in time*

$$(\log N)^2 \cdot L_B\left(\frac{1}{2}, \sqrt{2}\right),$$

where the factor $(\log N)^2$ comes from the multiplication of two N-bits integers.

1.4.4 Smooth ideals

For our purposes, we need to extend these results on smoothness to ideals.

Definition 1.4.19. For an integer $B \in \mathbf{N}$, we say that an ideal $\mathfrak{a}$ is B-smooth if all its prime factors have a norm below B.

Scourfield essentially shows in [Sco04] that the results of Dickman can be generalized to number fields. However, as in the case of integers, this does not suffice and we need a stronger assumption, which we formulate in the following way.

Heuristic 1.4.20. *The probability $\mathcal{P}(x,y)$ that an ideal of norm bounded by x is y-smooth satisfies*

$$\mathcal{P}(x,y) \geq e^{-u(\log u)(1+o(1))} \quad \textit{for} \quad u = \frac{\log x}{\log y}.$$

We stress that this is the exact correspondence of what have been proven for integers. This heuristic already appears in the work of Biasse and Fieker [BF14, Heuristic 1] about class group computation. Obviously, we also want to make use of the L-notation in context of number fields. We have to choose a quantity that represents the *size* of the number fields in order to express various quantities (probabilities, complexities, etc.) related to them. The natural choice is the absolute discriminant of the number field $|\Delta_{\mathbf{K}}|$. The previous heuristic then admits a neat rewriting in terms of the handy L-notation:

Corollary 1.4.21. *Assuming that $x = Ł_{|\Delta_{\mathbf{K}}|}(\alpha_1, c_1)$, $y = Ł_{|\Delta_{\mathbf{K}}|}(\alpha_2, c_2)$, and $\alpha_1 > \alpha_2$, Heuristic 1.4.20 can be expressed as*

$$\mathcal{P}(x,y) = L_{|\Delta_{\mathbf{K}}|}\left(\alpha_1 - \alpha_2, (\alpha_1 - \alpha_2)\frac{c_1}{c_2}\right)^{-1}.$$

Note that Seysen [Sey87] has proven in 1985 a similar result for quadratic number fields. For arbitrary degree, it remains conjectural, even under GRH.

Smoothness tests. Testing smoothness for ideals is not very complicated, assuming that we know how to test smoothness for integers. Indeed, given $B \in \mathbf{N}$, if $\mathfrak{a}$ is B-smooth, then in particular its norm $\mathcal{N}(\mathfrak{a})$ is B-smooth. Therefore, testing smoothness for ideals essentially amounts to testing smoothness for ideal norms. Computing the norm of an ideal is easy (see Section 1.4.2) and has a polynomial runtime in both the extension degree and the size of the norm. Once we know the prime numbers appearing in the norm, it suffices to find the valuations at the prime ideals above them. A way to figure out these valuations is explained in [Coh93, Section 4.8.3]. The algorithm described also has a complexity that is polynomial in the extension degree and the size of the prime number p.

Finally, the runtime of ideal smoothness tests is the same as integer smoothness tests:

$$\mathrm{Poly}\big(n, \log \mathcal{N}(\mathfrak{a})\big) \cdot L_B\left(\frac{1}{2}, \sqrt{2}\right),$$

where n is the extension degree of the field and $\mathcal{N}(\mathfrak{a})$ the norm of the ideal we want to test.

1.5 Ideal classes & Units

1.5.1 The Class Group

We now have all we require for giving the definition of the class group. It is defined as a set of equivalence classes:

Definition 1.5.1. Let $\mathbf{K}$ be a number field and $\mathcal{O}_\mathbf{K}$ its ring of integers. We say that two fractional ideals $\mathfrak{a}$ and $\mathfrak{b}$ in $\mathbf{K}$ are *equivalent* if there exists $x \in \mathbf{K}^*$ such that $\mathfrak{b} = \langle x \rangle \mathfrak{a}$. The set of equivalence classes is called the class group of $\mathcal{O}_\mathbf{K}$ — or $\mathbf{K}$ — and is denoted by $Cl(\mathcal{O}_\mathbf{K})$.

As the set of fractional ideals is a group, $Cl(\mathcal{O}_\mathbf{K})$ also is a group.

Theorem 1.5.2. *For any number field* $\mathbf{K}$*, the class group* $Cl(\mathcal{O}_\mathbf{K})$ *is a finite abelian group, whose cardinality, called the* class number*, is denoted by* $h_\mathbf{K}$.

We give the exact sequence that involves the class group

$$1 \longrightarrow \mathcal{P}(\mathbf{K}) \longrightarrow \mathcal{I}(\mathbf{K}) \longrightarrow Cl(\mathcal{O}_\mathbf{K}) \longrightarrow 1,$$

where $\mathcal{I}(\mathbf{K})$ denotes the set of ideals and $\mathcal{P}(\mathbf{K})$ the subset formed by the principal ones.

Remark 1.5.3. We can also define a class group for non-maximal orders in number fields. However, this complicates the definition because we have to limit to invertible ideals.

Computing the structure of the class group of a number field, and even only its class number, are major tasks in algorithmic number theory. This is the main problem I address during my PhD and this is the main topic of this thesis. A history of this domain is recalled in the following chapter.

1.5.2 The Group of Units

Class group computation is closely related to units in the number field. Thus we give basic results about units and briefly explain why we need them. By definition, a *unit* in $\mathbf{K}$ is an algebraic integer such that its inverse is also an algebraic integer. Equivalently, it is an algebraic integer of norm equal to ± 1.

Definition 1.5.4. The set of units in $\mathbf{K}$ is a multiplicative group, denoted by $\mathcal{U}(\mathbf{K})$. The torsion subgroup, formed of the so-called *roots of unity*, is denoted $\mu(\mathbf{K})$.

Again, we have an exact sequence

$$1 \longrightarrow \mathcal{U}(\mathbf{K}) \longrightarrow \mathbf{K}^* \longrightarrow \mathcal{P}(\mathbf{K}) \longrightarrow 1.$$

The main result about units in number fields is the *Dirichlet's Unit Theorem*:

Theorem 1.5.5. *Let (r_1, r_2) be the signature of* $\mathbf{K}$. *The group* $\mathcal{U}(\mathbf{K})$ *is a finitely generated abelian group of rank $r = r_1 + r_2 - 1$. We have a group isomorphism*

$$\mathcal{U}(\mathbf{K}) \simeq \mu(\mathbf{K}) \times \mathbf{Z}^r,$$

and $\mu(\mathbf{K})$ *is a finite cyclic group.*

From this theorem, we derive the existence of units $u_1, \dots, u_r$ such that every unit can be written in a unique way as $\zeta u_1^{n_1} \cdots u_r^{n_r}$ with $n_i \in \mathbf{Z}$ and ζ a root of unity. We call such a family (u_i) a *set of fundamental units*. Though this set is not unique, they all share the same kind of "determinant" because we go from one to another by multiplying by an $r \times r$ unimodular matrix. An issue arises when we tackle this kind of "determinant" as we are in the presence of a multiplicative group. That is why we introduce the *Log-unit lattice*.

Definition 1.5.6. The *logarithmic embedding* of $\mathbf{K}^*$ in $\mathbf{R}^{r_1+r_2}$ is the map Log that sends x to

$$\mathrm{Log}(x) = \big(\log|\sigma_1(x)|, \dots, \log|\sigma_{r_1}(x)|, 2\log|\sigma_{r_1+1}(x)|, \dots, 2\log|\sigma_{r_1+r_2}(x)|\big).$$

From this embedding, we directly derive the following statement.

Theorem 1.5.7. *The image of the group* $\mathcal{U}(\mathbf{K})$ *under the logarithmic embedding is a lattice of rank $r = r_1 + r_2 - 1$ in the hyperplane $\sum_{1 \le i \le r_1+r_2} x_i = 0$ of $\mathbf{R}^{r_1+r_2}$. It is called the* Log-unit lattice. *In addition, the kernel of this embedding is exactly equal to the group* $\mu(\mathbf{K})$ *of the roots of unity.*

Thanks to this logarithmic embedding, we have linearized our problem and we can now give the definition of this kind of "determinant".

Definition 1.5.8. The determinant of the Log-unit lattice is called the *regulator* of $\mathbf{K}$ and is denoted by $Reg_{\mathbf{K}}$. It can also be viewed as the absolute value of the determinant of any of the $r \times r$ matrices extracted from the $r \times (r+1)$ matrix

$$\big(\log\|\sigma_j(u_i)\|_{\mathbf{C}}\big)_{\substack{1 \le i \le r \\ 1 \le j \le r+1}}$$

for any set of fundamental units $u_1, \dots, u_r$. We use $\|x\|_{\mathbf{C}}$ to denote the absolute value of x if x is real and its square if x is complex (for the factor 2 appearing in the definition of the logarithmic embedding).

To be more precise, it is the calculation of — a close approximation of — the regulator that is linked with class group computation. On the other hand, computing the torsion subgroup

of $\mathcal{U}(\mathbf{K})$ is not that difficult. First, if $r_1 > 0$, the only roots of unity are ± 1. If $r_1 = 0$, it suffices to look for small vectors in the image through the canonical embedding of $\mathcal{O}_\mathbf{K}$. Cohen provides two algorithms for doing this in [Coh93, Section 4.9.2].

The Class Number Formula. Now we have defined all the notions, we sketch the link between class group and regulator.

Definition 1.5.9. Let $\mathbf{K}$ be a number field. We define for s with $\Re(s) > 1$ the Dedekind *zeta* function ζ_K of $\mathbf{K}$ by the formulae

$$\zeta_\mathbf{K}(s) = \sum_{\mathfrak{a}} \frac{1}{\mathcal{N}(\mathfrak{a})^s} = \prod_{\mathfrak{p}} \frac{1}{1 - \frac{1}{\mathcal{N}(\mathfrak{p})^s}},$$

where the sum is over all non-zero integral ideals of $\mathcal{O}_\mathbf{K}$ and the product is over all non-zero prime ideals of $\mathcal{O}_\mathbf{K}$.

Proposition 1.5.10. *Let $\mathbf{K}$ be a number field of degree $n = r_1 + 2r_2$. Let $w_\mathbf{K}$ denote the number of roots of unity. Then the function $\zeta_\mathbf{K}(s)$ converges absolutely for s with $\Re(s) > 1$ and extends to a meromorphic function defined for all complex s with only one simple pole at $s = 1$, whose residue satisfies*

$$\lim_{s \to 1} (s-1)\zeta_\mathbf{K}(s) = \frac{2^{r_1} \cdot (2\pi)^{r_2} \cdot h_\mathbf{K} \cdot Reg_\mathbf{K}}{w_\mathbf{K} \cdot \sqrt{|\Delta_\mathbf{K}|}}. \tag{1.18}$$

When we look carefully at this equality, we notice that on the right-hand side, all the factors can be easily computed, except for the product $h_\mathbf{K} Reg_\mathbf{K}$. Then, once we know an approximation of the left-hand side, it is easy to derive an estimation of this product. And this is the reason why class number and regulator are computed together. As explained in the remainder of this thesis, making use of this formula does not only allow to derive one from another, but this part is also mandatory for ensuring that we have correctly computed these values.

Another consequence of the Class Number Formula is the *Prime Ideal Theorem* stated by Landau in [Lan03]. It is the generalization of the Prime Number Theorem to number fields and provides an asymptotic formula for counting the number of prime ideals of bounded norm.

Theorem 1.5.11. *In every number field $\mathbf{K}$, the number of prime ideals of norm bounded by $N \in \mathbf{N}$, denoted by $\pi_\mathbf{K}(N)$, satisfies*

$$\pi_\mathbf{K}(N) \sim \frac{N}{\log N}.$$

Chapter 2

Previous work on class group computations and related problems

Contents

After the mathematical reminder in Chapter 1, we review the state of the art on class group computations together with related problems in number fields. Obviously, the first results obtained were for quadratic number fields.

2.1 Exponential strategies for quadratic number fields

Oddly, the very first result comes from Gauss, long before the birth of ideals.

2.1.1 Gauss and the equivalence classes of binary quadratic forms

Gauss was the first to compute the class number of quadratic number fields. However, that is not how he approached the problem: he was interested in *binary quadratic forms*.

We make use of the Proposition 1.2.12 for the representation of quadratics number fields as $\mathbf{K} = \mathbf{Q}(\sqrt{D})$, where D is the fundamental discriminant of the quadratic field.

Definition 2.1.1 ([Coh93, Definition 5.2.3])**.** A *binary quadratic form* f is a function $f(x,y) = ax^2 + bxy + cy^2$ where a, b and c are integers, which is denoted more briefly by (a,b,c). We say that f is *primitive* if $\gcd(a,b,c) = 1$. If f and g are two quadratic forms, we say that f and g are *equivalent* if there exists a matrix $M = \left(m_{i,j}\right) \in \mathrm{SL}_2(\mathbf{Z})$ — *i.e.*, an integral matrix of determinant equal to 1 — such that $g(x,y) = f(m_{1,1}x + m_{1,2}y, m_{2,1}x + m_{2,2}y)$.

This definition implies that *being equivalent* is an equivalence relation which preserves the discriminant $D = b^2 - 4ac$ of the quadratic form. In addition, if D is a fundamental discriminant, then any quadratic form of discriminant $D = b^2 - 4ac$ is primitive.

Before explaining how Gauss worked with these equivalence classes, we emphasize the link between ideals — and so class groups — in quadratic number fields and binary quadratic forms. Let D be a fundamental discriminant and let $\mathbf{QF}_D$ denote the set of equivalence classes of binary quadratic forms of discriminant equal to D, that is

$$\mathbf{QF}_D = \left\{(a,b,c) \mid b^2 - 4ac = D\right\}_{/SL_2(\mathbf{Z})}.$$

We then define the two functions

$$\begin{cases} \mathbf{QF}_D & \longleftrightarrow & Cl\left(\mathcal{O}_{\mathbf{Q}(\sqrt{D})}\right) \\ (a,b,c) & \overset{\phi_{FI}}{\longmapsto} & \left[a\mathbf{Z} + \frac{-b+\sqrt{D}}{2}\mathbf{Z}\right] \\ \frac{\mathcal{N}(x\omega_1 - y\omega_2)}{\mathcal{N}(I)} & \overset{\phi_{IF}}{\longmapsto} & [I] \\ & & \text{for } I = \omega_1\mathbf{Z} + \omega_2\mathbf{Z} \end{cases}$$

The first function ϕ_{FI} is well-defined, because it only depends on the orbit under $\mathrm{SL}_2(\mathbf{Z})$. However, it has a kernel, because the two forms (a,b,c) and $(-a,b,-c)$ define the same ideal class. Hence, the second function ϕ_{IF} cannot be well-defined: we have to fix an orientation for the basis (ω_1,ω_2), for instance we assume that $\frac{\omega_2\sigma(\omega_1)-\omega_1\sigma(\omega_2)}{\sqrt{D}} > 0$.

Theorem 2.1.2. *With the notation previously used,*

- *If $D<0$, we have a bijection between the class group $Cl\left(\mathcal{O}_{\mathbf{Q}(\sqrt{D})}\right)$ and the set*

$$\left\{(a,b,c) \mid b^2-4ac=D \text{ and } a>0\right\}\Big/_{SL_2(\mathbf{Z})}.$$

- *If $D>0$, we have a bijection between $\mathbf{QF}_D$ and $Cl\left(\mathcal{O}_{\mathbf{Q}(\sqrt{D})}\right)^+$, where this time we take the quotient only by principal ideals generated by an element of positive norm. This set is called the* narrow class group.

Since we have the exact sequence

$$1 \longrightarrow \{\pm 1\} \longrightarrow Cl(\mathcal{O}_{\mathbf{K}})^+ \longrightarrow Cl(\mathcal{O}_{\mathbf{K}}) \longrightarrow 1,$$

we conclude that in the real quadratic case $D>0$, the class number $h_{\mathbf{K}}$ is twice the number of orbits under $\mathrm{SL}_2(\mathbf{Z})$.

Now we have understood how we may derive the class number of a quadratic number field from the number of equivalence classes of binary quadratic forms, we can go back to the work of Gauss on these classes. For counting the number of equivalence classes, he focused on one representative for each one, namely the *reduced* form. Because variations appear between imaginary and real quadratic number fields, we split the study and begin examining the imaginary case.

Imaginary quadratic number fields. First assume that D is a fundamental discriminant satisfying $D<0$.

Definition 2.1.3. A positive definite quadratic form (a,b,c) is said to be *reduced* if $|b| \le a \le c$ and if, in addition, one of the two inequalities is an equality — *i.e.,* either $|b|=a$ or $a=c$ — then $b \ge 0$.

Proposition 2.1.4. *In every class of positive definite quadratic forms of discriminant $D<0$, there exists exactly one reduced form. Furthermore, if (a,b,c) is reduced, then $a \le \frac{\sqrt{|D|}}{3}$.*

Thanks to this result, it suffices to count the reduced forms of discriminant D such that $|b| \le a \le \frac{\sqrt{|D|}}{3}$ to obtain the class number of the quadratic field $\mathbf{Q}\left(\sqrt{D}\right)$. The algorithm is presented as Algorithm 1.

Algorithm 1 Counting reduced forms.

Input: A negative fundamental discriminant $D < 0$.
Output: The class number of binary quadratic forms of discriminant D.

```
1: Set h = 0, b ≡ D mod 2 and B = ⌊√|D| / 3⌋
2: while b ≤ B do
3:     Set q = (b^2 − D)/4 and a = b
4:     if a < 1 then
5:         Set a = 1
6:     end if
7:     while a^2 ≤ q do
8:         if a | q then
9:             if a = b or a^2 = q or b = 0 then
10:                Set h = h + 1
11:            else
12:                Set h = h + 2
13:            end if
14:        end if
15:        Set a = a + 1
16:    end while
17:    Set b = b + 2
18: end while
19: return h
```

This algorithm indeed counts the number of reduced forms of discriminant D. Its runtime is in $O(|D|)$, as it consists in two nested loops with $O\left(\sqrt{|D|}\right)$ terms.

Real quadratic number fields. Now assuming that $D > 0$, we have a similar process for real quadratic number fields.

Definition 2.1.5. Let $f = (a, b, c)$ be a quadratic form with positive discriminant D. We say that f is *reduced* if we have

$$\left|\sqrt{D} - 2|a|\right| < b < \sqrt{D}.$$

However, it is a little bit more complicated than in the imaginary case, because this time, every class contains a *cycle* of reduced forms. Then the class number of quadratic forms is exactly the number of such cycles. Here, *cycle* relates to the operation ρ defined as

$$\rho\big((a, b, c)\big) = \left(c, r, \frac{r^2 - D}{4c}\right),$$

where r is the unique integer such that $r \equiv b \bmod 2a$ and that satisfies $-|a| < r \leq |a|$ if $|a| > \sqrt{D}$ and $\sqrt{D} - 2|a| < r < \sqrt{D}$ otherwise.

Hence, there exists an algorithm for deriving the class number in the real case, as for the imaginary case, that runs in time $O(|D|)$ again. However, due to the difficulty of the process, we do not present this algorithm here.

2.1.2 Using analytic formulae

Another way to obtain the class number of quadratic — and even more general — number fields is to look at the Class Number Formula (Equation (1.18)). In case of quadratic fields, there exists an alternative form using *L-functions*, also introduced by Dirichlet[1]. For a fundamental discriminant D, they are defined as

$$L_D(s) = \sum_{n\geq 1} \left(\frac{D}{n}\right) \frac{1}{n^s},$$

and this series converges for $\Re(s) > 1$. In addition, we have

$$\zeta_{\mathbf{Q}(\sqrt{D})}(s) = \zeta(s) L_D(s). \tag{2.1}$$

The more essential point for using the Class Number Formula is that when $D < 0$, the group of units has rank 0, and so the regulator is 1. Eventually, this allows to recover directly the class number $h_{\mathbf{Q}(\sqrt{D})}$. However, the convergence of the L-function is not that fast and we require a large number of terms for having a good approximation.

Theorem 2.1.6 ([Coh93, Corollary 5.3.16]). *Let $D < -4$ be a fundamental discriminant. Then $h_{\mathbf{Q}(\sqrt{D})}$ is the closest integer to the sum*

$$\sum_{n=1}^{N} \left(\frac{D}{n}\right) \left(erfc\left(n\sqrt{\frac{\pi}{|D|}}\right) + \frac{\sqrt{|D|}}{\pi n} e^{-\pi n^2/|D|} \right),$$

where $erfc(x) = \frac{2}{\sqrt{\pi}} \int_x^\infty e^{-t^2}\, dt$ and $N = \left\lfloor \sqrt{\frac{|D|\log|D|}{2\pi}} \right\rfloor$.

Finally, we obtain a method for obtaining the class number of an imaginary quadratic number field that runs in time $O\left(|D|^{1/2+\varepsilon}\right)$ for any $\varepsilon > 0$. In the real case, *i.e.*, $D > 0$, there exists a similar result that allows to compute the class number in time $O\left(D^{1/2+\varepsilon}\right)$ for any $\varepsilon > 0$, assuming we know the regulator of the field. Indeed, this time, the group of units has rank 1 and the regulator is the logarithm of the unique generator greater than 1 of the torsion-free part of the unit group.

[1] These L-functions are different from the L subexponential notation introduced in Chapter 1.

Computing the regulator. We see that in order to obtain the class number, we have to find the regulator. Because we study the case of real quadratic number fields, the rank of the group of units is 1 and there is only one generator greater than 1. We denote it by u. We know that there exists a tuple $(a,b) \in \mathbf{Z}^2$ such that $u = \frac{a+b\sqrt{D}}{2}$. Because $\mathcal{N}(u) = 1$, we conclude that $a^2 - Db^2 = \pm 4$. We then obtain a *Diophantine equation* close to a *Pell equation*. More precisely, if $D \equiv 0 \bmod 4$, then it becomes equivalent to $a'^2 - \frac{D}{4}b^2 = \pm 1$. A good way to find a solution over $\mathbf{Z}$ is to use the *ordinary continued fraction expansion* of $\sqrt{D}$. That provides the solution within a number of steps in $O\left(D^{1/2+\varepsilon}\right)$ for any $\varepsilon > 0$. However, the size of a and b can be very large — such as $e^{\sqrt{D}}$ for instance — so that the full execution time is only bounded by $O\left(D^{1+\varepsilon}\right)$. This is not sufficient.

Keeping in mind that we are not really interested in the fundamental unit, but mostly in the regulator, we can modify the continued fraction algorithm for working with real numbers with a reasonable degree of accuracy. If the intermediate results are all computed to a given precision, we finally obtain an approximation of the regulator at this precision, with a runtime in $O\left(D^{1/2+\varepsilon}\right)$. This ends the proof that analytic methods can be used to compute the class number of any quadratic number field in time $O\left(|D|^{1/2+\varepsilon}\right)$ for any $\varepsilon > 0$.

2.1.3 Shanks' Baby-Step–Giant-Step method

The problem of computing class groups was addressed for the first time by Shanks in 1969. By computing class groups, we mean that we are interested in the group structure, and not only in its cardinality. Indeed, until now, all the described methods only provide the class number, but nothing about the structure.

In [Sha69], Shanks introduces a new method for deriving the class group structure of an imaginary quadratic number field in time $O\left(|D|^{1/4+\varepsilon}\right)$ for any $\varepsilon > 0$. This is the first occurrence in the literature of the now widespread *Baby-Step–Giant-Step method*, that has applications in various areas of number theory. His work brought two refinements, because it enables to obtain more information on the class group, and in less time.

The primary description is again for imaginary quadratic number fields. Let $D < 0$ be a fundamental discriminant. From the Class Number Formula (Equation (1.18)), we obtain a good enough approximation $\tilde{h}$ of the class number $h_{\mathbf{K}}$ by computing a partial sum of the Dirichlet function associated to D. Shanks claimed that 2^{17} terms generally suffice for having a relative error below 10^{-3}. For every small prime p such that the Legendre symbol satisfies $\left(\frac{D}{p}\right) = 1$, we set b_p as a square root of D mod p that has the same parity with D, *i.e.*, $D \equiv b_p \bmod 2$. Denoting by c_p the integer such that $b_p^2 - D = 4pc_p$, we thus obtain a quadratic form

$$f_p = \left(p, b_p, c_p\right) \tag{2.2}$$

of discriminant D, and we eventually have such a form for each small prime with $\left(\frac{D}{p}\right) = 1$.

Then his idea is to compute powers of a fixed quadratic form. Indeed, every form f satisfies $f^{h_{\mathbf{K}}} = \iota$, where ι denotes the *principal form*, defined according to the parity of D as

$$\iota = \left(1, 1, \frac{1-D}{4}\right) \quad \text{or} \quad \iota = \left(1, 0, \frac{-D}{4}\right).$$

We begin by computing $f^{\tilde{h}}$, with $\tilde{h}$ as above. This is performed using fast exponentiation together with compositions and reductions of quadratic forms. Then, we almost certainly have $f^{\tilde{h}} \neq \iota$ so most likely we require a correction term C such that $h_{\mathbf{K}} = \tilde{h} + C$. This value C may be assumed to be small compared to $h_{\mathbf{K}}$ (say, on the order of $\frac{h_{\mathbf{K}}}{1000}$), so it can be found efficiently using the Baby-Step–Giant-Step method. For an integer s to be determined, we compute and store the coefficients a_n, b_n of the first s powers of f:

$$f^n = (a_n, b_n, c_n) \quad \text{for } n \in \{1, \dots, s\}.$$

For free, we obtain a_n, b_n for $n \in \{0, -1, \dots, -s\}$ because $f^{-n} = (a_n, -b_n, c_n)$. Hence if we have $|h_{\mathbf{K}} - \tilde{h}| \leq s$, we find a match

$$f^{\tilde{h}} = (a_{\tilde{h}}, b_{\tilde{h}}, c_{\tilde{h}}) = (a_n, b_n, c_n) = f^n,$$

for some $n \in \{-s, \dots, s\}$. We only take care of the first two coefficients because as the discriminant is fixed, the third one is completely determined by the two others. If $|h_{\mathbf{K}} - \tilde{h}| > s$, we set $g = 2s$ and compute successively the power $f^{\tilde{h}+rg}$ for $r = 1, -1, 2, -2, \dots$ until for an r to be determined, we get a match

$$f^{\tilde{h}+rg} = f^n.$$

In the end, we have computed s baby-steps and $2|r|$ giant-steps until we have found an equality $f^{\tilde{h}+rg-n} = \iota$. The best parameters choice is $s = 2|r|$ and $r \times s \approx C$, that is r and s of order $\sqrt{C}$. As the class number satisfies $h_{\mathbf{K}} = O\left(|D|^{1/2+\varepsilon}\right)$, we get r and s about $O\left(|D|^{1/4+\varepsilon}\right)$, so that the full number of compositions we have to perform is also bounded by $O\left(|D|^{1/4+\varepsilon}\right)$, for any $\varepsilon > 0$.

In addition, we can recover the order of the quadratic forms and then infer the group structure. An issue arises when the order e of f is small, because it implies that other multiples of e are close to $\tilde{h}$. However, this can be fixed simply by looking at another form.

Despite the breakthrough provided by this new method, the result suffers from shortcomings in the complexity study. However, Lenstra [Len82] and Schoof [Sch82] independently proved in 1982 that the runtime of the algorithm can be reduced to $O\left(|D|^{1/5+\varepsilon}\right)$ for any $\varepsilon > 0$, assuming the *Generalized Riemann Hypothesis*.

Conjecture (Generalized Riemann Hypothesis (GRH))**.** *For every fundamental discriminant D, the L-function $L_D(s) = \sum_{n\geq 1} \left(\frac{D}{n}\right)\frac{1}{n^s}$ can be extended by analytic continuation to a meromorphic function defined on the whole complex plane. The* Generalized Riemann Hypothesis *asserts that all the zeros of L_D in the critical strip $0 \leq \Re(s) \leq 1$ lie on the critical line $\Re(s) = \frac{1}{2}$.*

Remark 2.1.7. To be accurate, GRH is defined for all *Dirichlet characters* χ and associated L-functions $L(\chi, s) = \sum_{n\geq 1} \frac{\chi(s)}{n^s}$. However, as we are only interested in the *Kronecker symbol* $\left(\frac{D}{n}\right)$ this statement suffices for our purposes.

At first, Lenstra and Schoof showed that computing the approximation $\tilde{h}$ with $O\left(|D|^{1/5+\varepsilon}\right)$ terms suffices to bound the relative error by

$$|h_{\mathbf{K}} - \tilde{h}| \leq B \qquad \text{with } B = O\left(|D|^{1/5+\varepsilon}\right).$$

This is a consequence of results from Lagarias, Odlyzko, and Oesterlé [LO77, Oes79] that requires the assumption of GRH. Hence, all the calculations performed have a runtime in $O\left(|D|^{1/5+\varepsilon}\right)$, for any $\varepsilon > 0$. In addition, still assuming GRH, Lagarias, Odlyzko, and Montgomery [LOM79] obtained a generating set of forms for the class group.

Lemma 2.1.8. *Assuming GRH, the class group of quadratic binary forms is generated by all the forms f_p for p primes such that $p \leq c_0 \left(\log|D|\right)^2$, for an effectively computable constant c_0.*

This lemma bounds the number of forms we have to use during the computations and ensures that the global complexity is $O\left(|D|^{1/5+\varepsilon}\right)$.

For the real case, Shanks introduced a similar method, using the *infrastucture*, that is a group-like structure appearing in real quadratic number fields (see [Fon09] for more details about infrastructure). Shanks claimed a runtime in $O\left(|D|^{1/4+\varepsilon}\right)$ as for the imaginary case, and Lenstra and Schoof also proved that it is in fact $O\left(|D|^{1/5+\varepsilon}\right)$, assuming GRH.

2.2 Class group generation

Before continuing, we need results about the generation of the class group for all number fields. The first result we want to cite is from Minkowski. He shows [PZ89, Corollary 2.9] that every class contains an ideal of norm not exceeding Minkowski's bound

$$M_{\mathbf{K}} = \left(\frac{4}{\pi}\right)^{r_2} \frac{n!}{n^n}\sqrt{|\Delta_{\mathbf{K}}|}. \tag{2.3}$$

This is a direct consequence of the following theorem, also due to Minkowski.

Theorem 2.2.1. *For every lattice $\mathcal{L} \subset \mathbf{R}^n$ and any convex set in $\mathbf{R}^n$ which is symmetric with respect to the origin and with volume greater than $2^n \cdot \det \mathcal{L}$, there exists a non-zero lattice point that belongs to the convex set.*

Unfortunately, this is not sufficient because an algorithm that uses $M_{\mathbf{K}}$ cannot have a subexponential complexity. This was the reason of the work of Lenstra [Len82] and Schoof [Sch82] for quadratic fields. Later, Bach [Bac90] extended this result to all number fields, under the assumption of ERH.

Conjecture (Extended Riemann Hypothesis (ERH))**.** *The* Extended Riemann Hypothesis *asserts that, for any Dedekind zeta-function $\zeta_{\mathbf{K}}(s)$, all the zeros of $\zeta_{\mathbf{K}}$ in the critical strip $0 \leq \Re(s) \leq 1$ lie on the critical line $\Re(s) = \frac{1}{2}$.*

Remark 2.2.2. In case of quadratic number fields, ERH and GRH are equivalent because of the equality of Equation (2.1). However, we introduce here ERH for the generalization to arbitrary degree number fields.

Theorem 2.2.3 ([Bac90, Theorem 4.4])**.** *Assuming ERH, the class group of a number field $\mathbf{K}$ of discriminant $\Delta_{\mathbf{K}}$ is generated by the prime ideals of norm at most $12\left(\log|\Delta_{\mathbf{K}}|\right)^2$. In the quadratic case, this bound can be reduced to $6\left(\log|\Delta_{\mathbf{K}}|\right)^2$.*

Note that this bound is not easily comparable with Minkowski's. One bound asserts the existence of a representative of the class with bounded norm whereas the other bound gives a bound for constructing a large enough set of generators. Belabas *et al.* [BDF08] also work on this generation bound and find some practical improvements.

Approximation of the residue. All these results also have consequences for the computation of — an approximation of — the Euler Product.

Buchmann and Williams [BW89] first understood that using averaging procedures permits to be more precise than truncated products. However, we had to wait for the extensive study of Bach [Bac95], who states that, assuming ERH, a good enough approximation $\tilde{h}$ of $h_{\mathbf{K}}Reg_{\mathbf{K}}$ such that $h_{\mathbf{K}}Reg_{\mathbf{K}} \leq \tilde{h} \leq 2h_{\mathbf{K}}Reg_{\mathbf{K}}$ can be computed in polynomial time in the discriminant of the field considering only the primes below $O\left(\log^2|\Delta_{\mathbf{K}}|\right)$.

2.3 Subexponential complexity, using index calculus method

The current best algorithms for class group computation rely on the index calculus method. It is also the case for factoring integers or computing discrete logarithms in finite fields. A brief summary is as follows:

1. Fix a factor base composed of small elements and that is large enough to generate all elements of the group.
2. Collect relations between those small elements, corresponding to linear equations.
3. Deduce the result sought performing linear algebra on the system built from the relations.

We give more details about the different steps in case of class group computation. Afterwards, every contribution is examined with respect to this global strategy.

The factor base. We define the factor base $\mathcal{B}$ as the set of all prime ideals in $\mathcal{O}_{\mathbf{K}}$ that have a norm bounded by a constant B. This bound must be chosen such that the factor base generates the whole class group. As recalled in Section 2.2, Bach has shown that assuming ERH, the classes of ideals with a representative of norm less than $12\left(\log|\Delta_{\mathbf{K}}|\right)^2$ suffice to generate the class group. However, as the ability to find relations in the collection step increases with the size of the factor base, we fix

$$B = Ł_{|\Delta_{\mathbf{K}}|}(\beta, c_b),$$

for values of β and c_b with $0 < \beta < 1$ and $c_b > 0$ that are determined later.

Thanks to the Landau Prime Ideal Theorem (see Theorem 1.5.11), we know that in every number field $\mathbf{K}$, the number of prime ideals of norm bounded by B, denoted by $\pi_{\mathbf{K}}(B)$, satisfies

$$\pi_{\mathbf{K}}(B) \sim \frac{B}{\log B}. \tag{2.4}$$

As a consequence, the cardinality of the factor base is about B, namely:

$$N = |\mathcal{B}| = L_{|\Delta_{\mathbf{K}}|}(\beta, c_b).$$

Relation collection. Let $\mathfrak{p}_i$, $1 \leq i \leq N$, denote the N prime ideals in the factor base $\mathcal{B}$. As their classes generate $Cl(\mathcal{O}_{\mathbf{K}})$, we have a surjective morphism $\phi : \mathbf{Z}^n \longrightarrow Cl(\mathcal{O}_{\mathbf{K}})$ via

$$\begin{array}{ccccc} \mathbf{Z}^N & \longrightarrow & \mathcal{I} & \longrightarrow & Cl(\mathcal{O}_{\mathbf{K}}) \\ (e_1, \cdots, e_N) & \longmapsto & \prod_i \mathfrak{p}_i^{e_i} & \longmapsto & \prod_i [\mathfrak{p}_i]^{e_i}, \end{array} \tag{2.5}$$

and the class group $Cl(\mathcal{O}_{\mathbf{K}})$ is then isomorphic to $\mathbf{Z}^N / \ker\phi$. By computing the kernel of this morphism, we deduce the class group, which is given by the lattice of the vectors $(e_1, \cdots, e_N)$ in $\mathbf{Z}^N$ for which $\prod \mathfrak{p}_i^{e_i} = \langle x \rangle$ with $x \in \mathbf{K}^*$. Thus the relations that we want to collect are given by x in $\mathbf{K}^*$ such that

$$\langle x \rangle = \prod \mathfrak{p}_i^{e_i}. \tag{2.6}$$

Relation collection is the main part of the algorithm, so that it is the one that has been significantly improved with the new contributions.

Linear algebra. Once the relations are collected, we store them in a matrix. A row corresponds to an algebraic number x and the i-th coefficient is the valuation of the principal ideal $\langle x \rangle$ at $\mathfrak{p}_i$ — that is e_i in Equation (2.6). These valuations e_i are computed by looking first at the norm of $\langle x \rangle$, as explained in Section 1.4.4. Then, the structure of the class group is given by the *Smith Normal Form* (SNF) of the matrix (see Section 1.1.1). More precisely, we first compute the *Hermite Normal Form* (HNF) with a pre-multiplier since we need kernel vectors in the verification step (as explained below). Finally, the class number can be deduced by multiplying the diagonal coefficients of the HNF while the group structure is given by the diagonal coefficients of the SNF.

Verification. The group H provided by the linear algebra step is only a candidate for the class group and has to be verified. Indeed, even assuming that the factor base is large enough to generate the full class group, the number of relations derived may be insufficient. In that case, the class group $Cl(\mathcal{O}_{\mathbf{K}})$ is only a quotient of the candidate H. Fortunately we can obtain some information on the class number from the Class Number Formula (Proposition 1.5.10). As mentioned in Section 2.2, a good approximation of the Euler Product can be calculated in order to get an estimation of the product $h_{\mathbf{K}} Reg_{\mathbf{K}}$.

Thus we need at least an approximation of the regulator of the number field in order to carry out this verification. Fortunately, it does not cost too much to determine a candidate for the regulator once we have our candidate for the class group. Indeed, the collected relations make it possible to infer one: by looking for elements in the kernel of the relation matrix, we are computing units of $\mathbf{K}$. Then once we have found generators of the group spanned by these units, it only remains to compute a determinant. If these generators form a set of fundamental units, we get the regulator. Otherwise, we have only found a multiple of the regulator, because the group spanned by those is a subgroup of the unit group $\mathcal{U}(\mathbf{K})$.

In the end, when we have the — hypothetical — class number and regulator, it is enough to compare their product with the approximation calculated from the Euler Product. Either the ratio is close to 1 in which case the two quantities are the correct ones, or it is not and more relations are required. This verification step works since both class number and regulator are computed decreasingly: if there is a sufficient number of primes ideals — respectively units — involved, then adding a relation can only reduce the class number — respectively the regulator — by an integer factor. As a consequence, the ratio is close to 1 only for $h_{\mathbf{K}}$ and $Reg_{\mathbf{K}}$.

2.3.1 First example: imaginary quadratic number fields, by Hafner and McCurley

Again, the initial subexponential result was obtained for imaginary quadratic number fields, by Hafner and McCurley. As the prior results, they did not use ideals, but binary quadratic forms. We recall that in those fields, the regulator is 1, because the rank of the unit group is zero. Note that the results of Bach stated in Section 2.2 were not released at that time. However, this is not a problem because for quadratic number fields, the analysis was done by Lenstra and Schoof.

The factor base $\mathcal{B}$ is fixed as the set of the forms $f_p = (p, b_p, c_p)$ — as defined in Equation (2.2) — for all primes $p \leq B = Ł_{|D|}\left(\frac{1}{2}, \frac{\sqrt{2}}{4}\right)$ such that $\left(\frac{D}{p}\right) = -1$. Thanks to Lemma 2.1.8, we know that this set of forms is large enough to generate the whole class group. In fact, it is much larger than required to generate the class group: as we want to decompose forms over this set efficiently, we need such a size. The construction of the factor base is naive: it relies on an enumeration of all the primes below the bound. It follows from the Chebotarev's density theorem [Che26] combined with Theorem 1.5.11 that the number of such primes can be expected to be $\frac{B}{2\log B} = L_{|D|}\left(\frac{1}{2}, \frac{\sqrt{2}}{4}\right)$. At the same time, we can use the enumeration of primes to find an approximation of $h_{\mathbf{K}}$. Using the techniques of Schoof [Sch82], we know that there exists a constant c_1 such that computing the partial sum of the L-function up to $c_1\left(\log|D|\right)^2$ suffices to find an approximation $\tilde{h}$ such that

$$\frac{3}{4} < \frac{\frac{22}{7} h_{\mathbf{K}}}{\lfloor\sqrt{D}\rfloor \tilde{h}} < \frac{3}{2}, \tag{2.7}$$

where $\frac{22}{7}$ is an approximation of π. Then, we easily derive a rational b that satisfies $b \leq h_{\mathbf{K}} \leq 2b$, which suffices for our purposes. The details are given in [McC89]. Both of the described parts of the algorithm run in time $O(|\mathcal{B}|) = L_{|D|}\left(\frac{1}{2}, \frac{\sqrt{2}}{4}\right)$.

Hafner and McCurley used that the class group can be described as the quotient of $\mathbf{Z}^N$ by a lattice — or a $\mathbf{Z}$-module — of relations as described in Equation (2.5). In their article, ideal products are replaced by quadratic-form compositions, but the structures are the same, as recalled in Theorem 2.1.2. The main part of the algorithm consists in the derivation of relations between elements of the factor base.

The generation of the relations is based on the work of Seysen described for integer factorization [Sey87]. For every form f_p in the factor base $\mathcal{B}$, we reduce the form[2]

$$f_p^{2N|D|} \prod_{i=1}^{N} f_{p_i}^{x_i},$$

[2]The reason for the exponent $2N|D|$ comes later in the section.

where the exponents x_i are chosen randomly from a uniform distribution on $\{0,\dots,|D|\}$ for the forms f_{p_i} with $p_i \leq c_0 \left(\log|D|\right)^2$ and $x_i = 0$ otherwise. Let (a,b,c) denote the reduced form in the class. If a is $Ł_{|D|}\left(\frac{1}{2},\frac{\sqrt{2}}{4}\right)$-smooth, then (a,b,c) splits completely over the factor base $\mathcal{B}$ and we obtain a relation

$$f_p^{2N|D|}\prod_{i=1}^{N} f_{p_i}^{x_i} = (a,b,c) = \prod_{j=1}^{N} f_{p_j}^{y_j}.$$

If the smoothness test fails, we modify the randomization product $\prod f_{p_i}^{x_i}$ and reduce the new form we get. This step is repeated until $N = |\mathcal{B}|$ relations have been collected, involving all N quadratic forms in the factor base.

For testing the smoothness of the reduced form, Hafner and McCurley choose to use trial divisions, which runs in time $O(N)$ — instead of using methods described in Section 1.4.4. The complexity of the relation collection also depends on the number of forms we have to test for smoothness. The estimate below was first presented by Seysen in the context of integer factorization.

Theorem 2.3.1 ([Sey87, Proposition 4.4])**.** *Assuming ERH, the number of reduced forms (a,b,c) such that a is B-smooth for $B = Ł_{|D|}\left(\frac{1}{2},\frac{\sqrt{2}}{4}\right)$ is at least $L_{|D|}\left(\frac{1}{2},\frac{\sqrt{2}}{2}\right)^{-1}$.*

Hence, to obtain one relation, we need to test about $L_{|D|}\left(\frac{1}{2},\frac{\sqrt{2}}{2}\right)$ forms; that leads to a total number of $L_{|D|}\left(\frac{1}{2},\frac{\sqrt{2}}{4}+\frac{\sqrt{2}}{2}\right)$ forms to find N relations. As each form requires at most N trial divisions, the final complexity of the collection step is

$$L_{|D|}\left(\frac{1}{2},\sqrt{2}\right).$$

At this point, we have a square matrix that is sparse and diagonally dominated — because of the exponents $2N|D|$. We compute its determinant modulo p for small values of primes p using Gaussian elimination and recover the determinant using the *Chinese Remainder Theorem* and the fact that it is upper bounded by $N^{\frac{5N}{2}}|D|^N$ — this last bound is derived from Hadamard's inequality and bounds on the exponents of our construction. The diagonally-dominated property implies that this matrix has rank N, so it corresponds to a sublattice of the lattice of relations. However, the relations we have used have a special form. Therefore it is unlikely that this sublattice is the whole lattice of relations.

Then, we need new relations, corresponding to elements chosen almost uniformly at random. So we look at forms $\prod f_{p_i}^{x_i}$ for random $x_i \in \{-|D|^2,\dots,|D|^2\}$, reduce them, and test their smoothness as we have previously done. It is claimed in [HM89] that N more relations of this form suffice to generate the whole lattice of relations with probability $1-\frac{1}{|D|}$. The determinant of the $N\times N$ matrix we have computed allows us to use modular arithmetic.

Again, the runtime for this relation collection is in

$$L_{|D|}\left(\frac{1}{2},\sqrt{2}\right).$$

Finally, we know that we have the whole lattice of relations — with probability $1-\frac{1}{|D|}$ — and it only remains to find the invariants of the class group. We begin by computing the Hermite Normal Form (HNF) of the $N \times 2N$ matrix built from the relations. This part is well explained in [HM89]. As the results of Storjohann recalled in Section 1.1.1 were not known at that time, they make use of the extended Euclidean algorithm to obtain a lower triangular matrix using unimodular column operations. All the calculations are performed modulo the determinant of the initial $N \times N$ matrix. Then the HNF is derived using similar techniques. The algorithms they use for linear algebra require to have a matrix diagonally dominated, that is why our matrix has this shape — for more details, see [HM89]. At this point, we know the class number $h_{\mathbf{K}}$: it is the product of the diagonal coefficients of the HNF. We can verify that it satisfies the inequality of Equation (2.7).

The structure is eventually provided by the Smith Normal Form of the HNF. Again, we make use of *gcd* calculations and unimodular column operations, this time performed modulo $h_{\mathbf{K}}$. All the linear algebra tools we use — determinant, HNF and SNF computations — run in time $N^{4+o(1)} = L_{|D|}\left(\frac{1}{2},\sqrt{2}\right)$ (see [HM89] for a justification for the exponent), so that it provides the final complexity of the complete algorithm. Note that the relation collection and the linear algebra steps present exactly the same complexity. That is no coincidence. Indeed, this is due to our choice for the size of the factor base — the second constant in the $L_{|D|}\left(\frac{1}{2}\right)$ — since the best complexity is obtained when there is a balance between the complexities of these two steps.

2.3.2 Buchmann extension to all number fields

In [Buc90], Buchmann extends the idea of Hafner and McCurley for arbitrary-degree number fields. No more binary quadratic forms are involved in the description, only ideals.

The way to derive the relations is similar to the collection performed for quadratic number fields, except that we do not require a *diagonally-dominated* matrix. Indeed, we build random ideals as power-products of elements of the factor base: $A = \prod \mathfrak{p}_i^{v_i}$. Then we find a *reduced* ideal A' in the class of A. This reduction hinges on finding a shortest non-zero vector in A which runs in polynomial time (see Section 1.1.3) — it is exponential in the degree but the

degree is fixed. Eventually if A' splits over the factor base $\mathcal{B}$, we have $A' = \prod \mathfrak{p}_i^{v'_i}$ so that

$$A \cdot (A')^{-1} = \prod_{i=1}^{N} \mathfrak{p}_i^{v_i - v'_i}$$

is principal, because A and A' are in the same class. This equality leads to a relation since we have found the vector $(v_1 - v'_1, \cdots, v_N - v'_N)$, which belongs to the kernel.

Fixing the smoothness bound $B = Ł_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}, c_b\right)$, and so the size of the factor base N, we need an estimate of the probability that a reduced ideal splits over the factor base. Buchmann then made the following assumption:

Heuristic 2.3.2. *The probability that a reduced ideal is $Ł_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}, c_b\right)$-smooth is $L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}, \frac{1}{4c_b}\right)^{-1}$.*

This is true for quadratic fields: Seysen proved it in the imaginary case and it was easily extended to the real case. This is also a consequence of Heuristic 1.4.20 and Corollary 1.4.21. Indeed, reduced ideals have norm bounded by $L_{|\Delta_\mathbf{K}|}\left(1, \frac{1}{2}\right)$ and the expression of the second constant leads to $\frac{1}{4c_b}$.

We also require a bound for the number of relations we need. The strategy followed by Buchmann assumes that:

Heuristic 2.3.3. *$L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}, c_b\right)$ relations suffice to find a generating system of the lattice of relations.*

The complexity of the collection phase is identical as the quadratic case: for getting $L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}, c_b\right)$ relations, Heuristic 2.3.2 reveals that testing $L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}, c_b + \frac{1}{4c_b}\right)$ ideals is necessary. As every test requires $L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}, c_b\right)$ operations — still using trial divisions — we obtain a final complexity in

$$L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}, 2c_b + \frac{1}{4c_b}\right).$$

Note that the extension degree of the field has to be fixed, as solving the SVP is polynomial in the size of the entries but exponential in the dimension. That is why we only can obtain a subexponential complexity *for a fixed degree*.

In addition, as we have already explained, it is appropriate to calculate the regulator simultaneously. Hence, we rather consider a lattice of dimension $N + r$, where N is the cardinality of the factor base and $r = r_1 + r_2 - 1$ the rank of the unit group. Naturally, the r last coordinates are given by the logarithms of the complex embeddings of the principal-ideal generator, which corresponds to the shortest vector identified in the reduction phase. Finally, this construction implies that the determinant of the whole lattice of relations is $h_\mathbf{K} Reg_\mathbf{K}$.

The linear algebra step is also more complicated. If we can figure out the HNF and SNF of the integer part of the relation matrix in time $O(N^4)$, it seems that the computation of the

regulator require $O(N^9)$ operations — because of precision issues. Then, the global complexity is obtained when the equality $2c_b + \frac{1}{4c_b} = 9c_b$ holds, that is for $c_b = \frac{\sqrt{7}}{14} \approx 0.189$. This means that the expected runtime of the whole algorithm is $L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}, 1.7\right)$.

We remark that at this point, we have not verified if the result is correct. To do that, we require more information about the product $h_\mathbf{K} Reg_\mathbf{K}$. All the previous approximations obtained thanks to the Class Number Formula are derived from Dirichlet L-functions, that are equivalent to *zeta*-functions for quadratic number fields. Using the results of Bach recalled in Section 2.2 — even though they came later — we get a way to ensure that the computed result is correct.

2.3.3 Release the degree, by Biasse and Fieker

In [BF14], Biasse and Fieker modify the reduction algorithm: they use BKZ-reductions, offering a trade-off between the time spent in the reduction and the approximation factor of the short vectors. Thus they reach a complexity which allows both the degree and the discriminant of the field to tend to infinity. In addition, they replace trial divisions by more efficient methods such as ECM for the smoothness tests (see Section 1.4.4). That allows to reduce the complexity of the relation collection by a factor of order N.

For the linear algebra phase, they provide a deep study on the precision in [BF14, Section 4]. Based on the HNF computation recalled in Theorem 1.1.3, they show that assuming we have a full-rank matrix of relations, we can derive the class group structure and an approximation of the regulator — and so verify our results — in time $O(N^{\omega+1})$ where N is the size of the factor base. Precision questions are handled using a p-adic approach, whereas another method using rational approximations is developed in [Bia14a]. Finally, they also give a way to provide a set of fundamental units, in *compact representation*.

We make use of their analysis on precision in the sequel. We do not give a detailed description of the relation collection here as we recall and improve it in Chapter 4. Biasse and Fieker achieved a $L_{|\Delta_\mathbf{K}|}\left(\frac{2}{3} + \varepsilon\right)$ complexity in the general case and $L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}\right)$ when the extension degree n satisfies the inequality $n \leq (\log|\Delta_\mathbf{K}|)^{3/4-\varepsilon}$.

We leave aside for the moment their $\mathfrak{q}$-descent strategy, recalled in Chapter 3. It consists in an improvement that allows to reduce the complexity to $L_{|\Delta_\mathbf{K}|}(a)$ with $\frac{1}{3} \leq a \leq \frac{1}{2}$ when the number field is defined by a polynomial having small height [BF14, Theorem 6.2].

2.4 Algorithms related to number fields

2.4.1 Reduction of defining polynomials, by Cohen and Diaz y Diaz

We have already mentioned in Section 1.2.1 that the number of defining polynomials for a given number field is infinite. For simplicity, it seems that the best choice is the polynomial having the smallest coefficients — *i.e.,* smallest height — since the degree is fixed. In addition, it allows more efficient computations in the number field. A particularly relevant fact is that when bounding the norm of algebraic integers in $\mathcal{O}_{\mathbf{K}}$, the height of the defining polynomial of the field appears in the bound (see Lemma 1.4.4 and Corollary 1.4.7).

Cohen and Diaz y Diaz in [CD91] present an algorithm for reducing defining polynomials, also mentioned in [Poh93, Chapter V]. In the sequel, we refer to it as the algorithm of Cohen and Diaz y Diaz.

They define the *size* of a monic degree-n polynomial $T = \prod_{j=1}^{n} (X - \tau_j)$ as

$$S(T) = \sum_{j=1}^{n} |\tau_j|^2$$

and it is the quantity they choose to minimize. Since for all $k \in \{0, \ldots, n\}$, the $(n-k)$-th coefficient satisfies

$$|t_{n-k}| \leq \binom{n}{k} \left(\frac{S(T)}{n}\right)^{\frac{k}{2}},$$

the size of T is related to the size of $\max |t_{n-k}|^{\frac{2}{k}}$ and hence to the height of the polynomial.

The main reason for this choice is that the size is not too hard to minimize. Indeed, computing a polynomial with the smallest size is equivalent to finding T such that the vector formed of its roots has the smallest L_2-norm. Remembering that this vector belongs to the lattice formed of the canonical embedding $\sigma(\mathcal{O}_{\mathbf{K}})$ in $\mathbf{R}^{r_1} \times \mathbf{C}^{r_2} \simeq \mathbf{R}^n$, this can be done using a lattice reduction algorithm. In particular, the LLL algorithm is a good option since it finds a basis of short vectors. However, while it solves the problem for small values of n, when the dimension grows, it is not guaranteed to find the shortest vector. If needed, it is possible to use stronger lattice reduction algorithms such as BKZ for instance.

Thus, the first step of method of Cohen and Diaz y Diaz is to compute an integral basis $\omega_1, \ldots, \omega_n$ of $\mathcal{O}_{\mathbf{K}}$, so that every algebraic integer $x \in \mathcal{O}_{\mathbf{K}}$ can be expressed as

$$x = \sum_{i=1}^{n} x_i \omega_i,$$

where the x_i are in $\mathbf{Z}$. Assuming that x generates $\mathbf{K}$, then its minimal polynomial coincides

with its characteristic polynomial

$$P_x = \prod_{j=1}^{n} \left(X - \sum_{i=1}^{n} x_i \sigma_j(\omega_i) \right).$$

This turns the correspondence between algebraic integers of degree n and monic polynomials defining the field $\mathbf{K}$ into a correspondence between polynomials and vectors in the lattice of embeddings of $\mathcal{O}_\mathbf{K}$ in $\mathbf{C}^n$. This lattice is generated by the n vectors:

$$\Omega_i = [\sigma_1(\omega_i), \ldots, \sigma_n(\omega_i)]\,. \tag{2.8}$$

When the field $\mathbf{K}$ is not primitive, care should be taken to avoid polynomials that generate a strict subfield[3] of $\mathbf{K}$. For an integer x, the size of its polynomial P_x is given by a positive definite quadratic form in the x_i:

$$S(P_x) = \sum_{j=1}^{n} \left| \sum_{i=1}^{n} x_i \sigma_j(\omega_i) \right|^2 = \sum_{i,j} \left(\sum_{k=1}^{n} \sigma_k(\omega_i) \overline{\sigma_k(\omega_j)} \right) x_i x_j.$$

Equivalently, one can remark that the coefficients $\sum_{k=1}^{n} \sigma_k(\omega_i)\overline{\sigma_k(\omega_j)}$ are the entries of the Gram matrix of the above lattice. Since lattice reduction, including the LLL algorithm, only requires this Gram matrix as input (see the work of Espitau and Joux in [EJ17]), one can find a basis of short vectors in $\mathbf{Z}^n$ corresponding to small values of $S(P_x)$. Finally, the algorithm of Cohen and Diaz y Diaz selects the best value for x it can find, breaking ties in any arbitrary manner when several values are equally good.

From an implementation perspective, the lattice generated by vectors Ω_i (and/or the Gram matrix of the lattice) cannot be represented exactly. Instead, it is explained in [Coh93] that, in general, it should be replaced by an approximation with sufficiently high precision. In fact, there is an interesting special case that occurs when $\mathbf{K}$ is totally real. In that case, the Gram matrix is integral and the lattice reduction can thus be performed on the exact lattice, represented by its Gram matrix. For the general case, an analysis of the necessary precision is given in [Bel04] and practical results are provided in [EJ17].

To summarize, the existing algorithm reduces the above lattice in order to find a primitive integer of $\mathbf{K}$ with smallest size. However, as already noticed by Cohen in [Coh93, Remark Algorithm 4.4.12], having the smallest size does not necessarily imply having the smallest height.

[3] This issue is taken care of in practice in `PARI/GP` by restricting the enumeration in order to avoid elements that only generate subfields — they have a minimal polynomial whose degree is strictly lower than n. In theory, since they can be an extremely large number of short bad elements, bounding the resulting complexity is difficult.

2.4.2 The Number Field Sieve

To finish, we also decide to briefly introduce the *Number Field Sieve* (NFS), the most efficient algorithm for integer factorization, as it uses methods that are close to the ones we use in class group computations. It also has applications in discrete logarithm computation in finite fields. The main part consists in the search for algebraic integers of smooth norms in number fields. This is a direct generalization of the *quadratic sieve*, where we extend the degree of the number field involved.

More precisely, for factoring an integer N, we introduce two irreducible polynomials $f, g \in \mathbf{Z}[X]$ of small degrees, that have a common root m when interpreted as polynomials modulo N. Selecting these polynomials can be done in many ways, and finding the best one is still a open problem. An easy choice is to fix the degree d together with the root m, and then fix f as $f(X) = \sum_{i=0}^{d} b_i X^i$ where the coefficients b_i are chosen in $\{0, \dots, m-1\}$ such that $N = \sum b_i m^i$ and $g(X) = X - m$. Thus f represents the expansion of N in base m. The next part involves a search of smooth values for $b^{\deg(f)} f(\frac{a}{b})$ and $b^{\deg(g)} g(\frac{a}{b})$ where a and b are chosen small in $\mathbf{Z}$. The values $b^{\deg(f)} f(\frac{a}{b})$ and $b^{\deg(g)} g(\frac{a}{b})$ correspond to the norms of the algebraic integers $a - r_f b \in \mathbf{Z}[r_f]$ and $a - r_g b \in \mathbf{Z}[r_g]$, where r_f and r_g denote roots of f and g. The way to find such pairs $(a, b) \in \mathbf{Z}^2$ is in a certain sense similar to what we expect for class group computation. Indeed, from such a pair (a, b) we derive a principal ideal $\langle a - r_f b \rangle$ in $\mathbf{Z}[r_f]$ that splits over the set of prime ideals of small norm.

To end our explanation of the Number Field Sieve, once many pairs (a, b) are collected, we can combine them into products of smooth norms that are squares both in $\mathbf{Z}[r_f]$ and $\mathbf{Z}[r_g]$. This step relies on Gaussian elimination. Then, because both ring $\mathbf{Z}[r_f]$ and $\mathbf{Z}[r_g]$ inject in $\mathbf{Z}/N\mathbf{Z}$, we obtain two squares that are equal to $a - mb \bmod N$. Eventually, we recover a relation of the form $x^2 \equiv y^2 \bmod N$, from which we may get a factor of N by finding the greatest common divisor of N and $y - x$.

This approach, applied in the context of class group computations, is addressed by Biasse in [Bia14a]. Though he claims being inspired by Enge, Gaudry and Thomé [EG07, EGT11] who reached an $L_{q^g}\left(\frac{1}{3}\right)$ complexity for computing discrete logarithm in Jacobians of genus g curves over $\mathbf{F}_q$ for low-degree curves, all those results are derived from the NFS. The main difference is that unlike for NFS, we cannot choose the number field we work with. Biasse describes in [Bia14a] an $L_{|\Delta_\mathbf{K}|}\left(\frac{1}{3}\right)$ algorithm for class group computation in very restrictive classes of number fields. We describe and extend his work in Chapter 5.

Part II

Reducing the complexity of Class Group Computation

Chapter 3

Reduction of the defining polynomial

Contents

To begin, the best complexities — below $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)$ — are obtained for number fields defined by a good polynomial, thanks to the $\mathfrak{q}$-descent algorithm of [BF14]. Thus our goal is to derive such good defining polynomials. We introduce a new algorithm that given a number field, outputs a defining polynomial whose height is minimal. This algorithm is exponential in the degree, and so is only attractive when the extension degree is not too large. However, considered as a precomputation, it allows to extend the conditional result of Biasse and Fieker to all the fields for which there exists a good polynomial. Indeed, even if we do not know it, thanks to our algorithm, we can now find it.

3.1 Motivations and link with class group computation

As we have illustrated with Lemma 1.4.4 and Corollary 1.4.7, it is mostly better to work in a number field that is defined by a good polynomial. As the degree is fixed, it seems reasonable to look for one having the smallest coefficients. In view of the results stated in [BF14], finding such a good defining polynomial may speed up the computations performed to determine class groups. Indeed, they propose the first algorithm with complexity below $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)$ for class group computation, when the number field is defined by a polynomial with small enough coefficients. More precisely, given parameters $n_0, d_0 > 0$ and $0 < \alpha < \frac{1}{2}$, they introduced the classes $\mathcal{C}_{n_0,d_0,\alpha}$ of number fields $\mathbf{K} \simeq \mathbf{Q}[X]/\langle T\rangle$ such that the defining polynomial $T \in \mathbf{Z}[X]$ of $\mathbf{K}$ satisfies

$$\begin{aligned} n = \deg T &= n_0(\log|\Delta_{\mathbf{K}}|)^{\alpha}(1+o(1)) \qquad \text{and} \\ d = \log H(T) &= d_0(\log|\Delta_{\mathbf{K}}|)^{1-\alpha}(1+o(1)). \end{aligned} \tag{3.1}$$

To be more specific, their results are slightly more general than what we state, since they consider the computation of class groups for all orders in $\mathbf{K}$. For simplicity, we only look at the maximal order $\mathcal{O}_{\mathbf{K}}$.

For these classes, they state the following theorem:

Theorem 3.1.1 ([BF14, Theorem 6.2])**.** *Under ERH and smoothness heuristics, if the number field* $\mathbf{K}$ *is in a class* $\mathcal{C}_{n_0,d_0,\alpha}$*, then there exists an* $L_{|\Delta_{\mathbf{K}}|}(a,c)$ *algorithm for class group and unit group computation for some* $c > 0$ *and a such that* $1-a \geq \alpha \geq 1-2a$*,* $a \geq \frac{1}{3}$ *and* $a \geq \alpha$*.*

The smoothness heuristic they use for this result is essentially similar to our Heuristic 1.4.20, except that they extend it to principal ideals that split over degree-1 prime ideals. In addition they require another heuristic to bound the number of relations they have to derive. The latter heuristic is stated later in the following section as Heuristic 4.2.2.

When a number field $\mathbf{K}$ is in $\mathcal{C}_{n_0,d_0,\alpha}$, it is defined by a polynomial T which verifies Equation (3.1), so that

$$\log H(T) = \frac{d_0 n_0 (1+o(1))}{n} \log|\Delta_{\mathbf{K}}|, \qquad \textit{i.e.,}\ H(T) = |\Delta_{\mathbf{K}}|^{\frac{\kappa}{n}} \tag{3.2}$$

where $\kappa = n_0 d_0 (1+o(1))$ tends to a constant.

We know from Proposition 1.2.16 that the height of T satisfies $H(T) \geq n^{\frac{n}{n-1}} |\Delta_{\mathbf{K}}|^{\frac{1}{2(n-1)}}$. This means that the best — *i.e.,* the smallest — value for κ we can hope for is $\kappa = \frac{1}{2} + o(1)$. For a randomly chosen polynomial, our experiments show that we may expect to be close to this best case.

However, as far as we know, the best theoretical bound on the smallest-height defining polynomials of an arbitrary number field is far from being that good. Indeed, it is a theorem from Hunter:

Theorem 3.1.2. *Let* $\mathbf{K}$ *be a number field of degree* n *and discriminant* $\Delta_{\mathbf{K}}$. *There exists* $\theta \in \mathcal{O}_{\mathbf{K}} \setminus \mathbf{Z}$ *whose minimal polynomial* P_θ *satisfies*

$$H(P_\theta) \leq 3^n \left(\frac{|\Delta_{\mathbf{K}}|}{n}\right)^{\frac{n}{2n-2}}.$$

In particular, this result provides a bound for κ in the case of primitive number fields, *i.e.,* number fields that do not contain non-trivial subfields. In this case, every element of $\mathcal{O}_{\mathbf{K}} \setminus \mathbf{Z}$ corresponds to a generator of $\mathbf{K}$ whose minimal polynomial defines $\mathbf{K}$.

Corollary 3.1.3. *In primitive number fields, there exists a defining polynomial* T *whose height satisfies*

$$H(T) \leq 3^n \left(\frac{|\Delta_{\mathbf{K}}|}{n}\right)^{\frac{n}{2n-2}}.$$

Note that this bound is much worse than we would like for our application. Indeed the bound on the height is only about the square root of the discriminant while we would like something of the order of the n-th root of the discriminant. Alternatively, we see that it corresponds to a value $\kappa \approx \frac{n}{2}$, much larger than the best case $\kappa \approx \frac{1}{2}$.

Proof of Theorem 3.1.2. Given the integral basis $\omega_1, \ldots, \omega_n$ of $\mathcal{O}_{\mathbf{K}}$ produced by the *Round 2 algorithm* [Coh93, Algorithm 6.1.8], we consider the lattice $\mathcal{L}$ generated by $(\Omega_1, \ldots, \Omega_n)$, where Ω_i denotes the vector $\sigma(\omega_i) = [\sigma_1(\omega_i), \ldots, \sigma_n(\omega_i)]$. We know that Ω_1 is the all ones vector and that $\|\Omega_1\| = \sqrt{n}$. In other words, Ω_1 corresponds to the integer 1 in the number field $\mathbf{K}$ and generates the trivial subfield $\mathbf{Q}$. Thus, elements of $\mathcal{O}_{\mathbf{K}} \setminus \mathbf{Z}$ are in a one-to-one correspondence with lattice vectors not on the line through 0 and Ω_1 — *i.e.,* with vectors that involve at least one of $\Omega_2, \ldots, \Omega_n$ with a non-zero coefficient.

We now consider the lattice $\mathcal{L}^\perp$ spanned by the vectors $\Omega_i^\perp = \Omega_i - k_i\Omega_1$, where $k_i = \frac{<\Omega_1,\Omega_i>}{n}$ so that $<\Omega_1,\Omega_i^\perp> = 0$. Hence, $\mathcal{L}^\perp$ is the orthogonal projection of $\mathcal{L}$ with respect to Ω_1, and its determinant satisfies

$$\det\mathcal{L} = \|\Omega_1\|\det\mathcal{L}^\perp.$$

In this new lattice $\mathcal{L}^\perp$, whose dimension is $n-1$, Minkowski's Theorem (Theorem 2.2.1) implies the existence of an element $\theta^\perp$ of small uniform norm, namely

$$\exists\theta^\perp \in \mathcal{L}^\perp \quad \text{such that} \quad \|\theta^\perp\|_\infty \leq \left(\det\mathcal{L}^\perp\right)^{\frac{1}{n-1}}.$$

Let us denote this vector by

$$\theta^\perp = \sum_{i=2}^{n} t_i\Omega_i^\perp = \sum_{i=2}^{n} t_i\Omega_i - \sum_{i=2}^{n} t_i k_i\Omega_1,$$

with $t_i \in \mathbf{Z}$.

Consider the affine line going through $\sum_{i=2}^{n} t_i\Omega_i$ and in the direction of Ω_1. On this line, there exists a lattice vector $\theta = \sum_{i=1}^{n} t_i\Omega_i$ of minimal length. We know that $\theta - \theta^\perp = \lambda\Omega_1$ for a real number λ with $|\lambda| \leq \frac{1}{2}$. As a consequence, $\|\theta - \theta^\perp\|_\infty \leq \frac{1}{2}$. This allows us to conclude that

$$\exists\theta \in \mathcal{L} \quad \text{such that} \quad \|\theta\|_\infty \leq \left(\frac{\det\mathcal{L}}{\sqrt{n}}\right)^{\frac{1}{n-1}} + \frac{1}{2} = \left(\frac{|\Delta_\mathbf{K}|}{n}\right)^{\frac{1}{2n-2}} + \frac{1}{2}.$$

Finally, thanks to the Minkowski's bound, we know that $|\Delta_\mathbf{K}| \geq n$ so that $\left(\frac{|\Delta_\mathbf{K}|}{n}\right)^{\frac{1}{2n-2}} \geq 1$ and $\left(\frac{|\Delta_\mathbf{K}|}{n}\right)^{\frac{1}{2n-2}} + \frac{1}{2} \leq \frac{3}{2}\left(\frac{|\Delta_\mathbf{K}|}{n}\right)^{\frac{1}{2n-2}}$. Then we deduce a — rough — upper bound for the height of the minimal polynomial P_θ of θ, using Equation (1.6) and $M(T) = \prod\max(1,|\tau_i|) \leq (\max|\tau_i|)^n$:

$$H(P_\theta) \leq 2^n\left(\frac{3}{2}\left(\frac{|\Delta_\mathbf{K}|}{n}\right)^{\frac{1}{2n-2}}\right)^n = 3^n\left(\frac{|\Delta_\mathbf{K}|}{n}\right)^{\frac{n}{2n-2}}.$$

□

For our purposes, it is difficult to directly work with the classes $\mathcal{C}_{n_0,d_0,\alpha}$. Instead, we introduce a more convenient and more general variation, associated with the L-notation. To be precise, we first introduce slightly modified classes $\mathcal{C}'_{n_0,d_0,\alpha}$, where we replace Equation (3.1) by

$$\begin{aligned} n = \deg T &= n_0\left(\frac{\log|\Delta_\mathbf{K}|}{\log\log|\Delta_\mathbf{K}|}\right)^\alpha(1+o(1)) \qquad \text{and} \\ d = \log H(T) &= d_0(\log|\Delta_\mathbf{K}|)^{1-\alpha}(\log\log|\Delta_\mathbf{K}|)^\alpha(1+o(1)). \end{aligned}$$

Introducing powers of $\log\log|\Delta_{\mathbf{K}}|$ in this way is more consistent with what is usually done for discrete logarithms computations using index calculus and simplifies the asymptotic analysis of index calculus algorithms which uses L-notation. All the theorems of [BF14] can be readily adapted to account for this change.

This modification being done, we now generalize the definition to include more number fields. Let $n_0 > 1$ be a real parameter arbitrarily close to 1, $d_0 > 0$, $\alpha \in [0,1]$ and $\gamma \in [0,1]$ such that $\alpha + \gamma \geq 1$, we define $\mathcal{D}_{n_0,d_0,\alpha,\gamma}$ as the set of all number fields $\mathbf{K}$ of discriminant $\Delta_{\mathbf{K}}$ that admit a monic defining polynomial $T \in \mathbf{Z}[X]$ of degree n that satisfies

$$\begin{aligned} \frac{1}{n_0}\left(\frac{\log|\Delta_{\mathbf{K}}|}{\log\log|\Delta_{\mathbf{K}}|}\right)^{\alpha} \quad &\leq \quad n \quad \leq \quad n_0\left(\frac{\log|\Delta_{\mathbf{K}}|}{\log\log|\Delta_{\mathbf{K}}|}\right)^{\alpha} \qquad \text{and} \\ d = \log H(T) \quad &\leq \quad d_0(\log|\Delta_{\mathbf{K}}|)^{\gamma}(\log\log|\Delta_{\mathbf{K}}|)^{1-\gamma}. \end{aligned} \tag{3.3}$$

For simplicity, $\mathcal{C}$, $\mathcal{C}'$, and $\mathcal{D}$ are often used for $\mathcal{C}_{n_0,d_0,\alpha}$,$\mathcal{C}'_{n_0,d_0,\alpha}$, and $\mathcal{D}_{n_0,d_0,\alpha,\gamma}$. In addition to the modification from $\mathcal{C}$ to $\mathcal{C}'$, there are two essential differences between our classes $\mathcal{D}$ and the previous classes $\mathcal{C}$ and $\mathcal{C}'$. First, we have two different exponents α and γ for $\log|\Delta_{\mathbf{K}}|$ in n and d, rather than using $\gamma = 1-\alpha$. Second, instead of having a $(1+o(1))$, we introduce inequalities in order to be more accurate. We also get the useful property that $\mathcal{D}_{n_0,d_0,\alpha,\gamma} \subset \mathcal{D}_{n_0',d_0',\alpha,\gamma'}$ whenever $n_0 \leq n_0'$, $d_0 \leq d_0'$ and $\gamma \leq \gamma'$.

Thanks to Corollary 3.1.3, we only need to consider classes $\mathcal{D}_{n_0,d_0,\alpha,\gamma}$, with γ at most 1. Moreover, the defining polynomials that appear in [BF14] essentially correspond to the lower end of the classes $\mathcal{D}$, where $\alpha+\gamma = 1$. Indeed, considering the modified classes $\mathcal{C}'$, we see that every number field $\mathbf{K}$ in $\mathcal{C}'_{n_0,d_0,\alpha}$ also belongs to $\mathcal{D}_{n_0,d_0,\alpha,1-\alpha}$. Thus, our broader definition of the classes includes more number fields, and not only those whose minimal defining polynomials satisfy $\alpha+\gamma = 1$.

Note that, in the context of class group computations, algorithms with complexity higher than $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)$ are not of great interest. As a consequence, since the algorithm we propose is exponential in n, we can limit our scope to classes with $\alpha \leq \frac{1}{2} \leq \gamma$.

3.2 An optimal algorithm for NF defining polynomial reduction

Now we know that existing bounds do not guarantee, for an arbitrary number field, the existence of a defining polynomial with coefficients small enough to allow application of Theorem 3.1.1, we thus aim at the next best thing: given an arbitrary defining polynomial for a number field, try to find a monic polynomial of smallest height that defines the same number field. We restrict ourselves to monic polynomials as they are minimal polynomials of algebraic integers.

Throughout this section, we assume that we have as input a monic irreducible polynomial T_{in} and a factorization of the discriminant $\Delta(T_{in})$ into prime factors. This allows us to precompute the discriminant $\Delta_{\mathbf{K}}$ of the number field $\mathbf{K}$ defined by T_{in} and an integral basis $\omega_1,\dots,\omega_n$ of $\mathcal{O}_{\mathbf{K}}$ using the techniques described in Section 1.2.4.

Since our goal is to find a monic polynomial of smallest height that defines $\mathbf{K}$, and such a polynomial clearly exists[1], let T_F denote one of them. Though uniqueness of T_F is not guaranteed — for instance $T_F(-X)$ also defines $\mathbf{K}$ and there might be others — any of these polynomials achieves our goal and more importantly, the minimal height $H(T_F)$ is well-defined. Let θ_F be a root of T_F; since T_F is monic, θ_F belongs to $\mathcal{O}_{\mathbf{K}}$. Thus, we can write

$$T_F(X) = \prod \big(X - \sigma_j(\theta_F)\big).$$

As Cohen and Diaz y Diaz did for the algorithm described in Section 2.4.1, we remark that the vector $\sigma(\theta_F)$ in the ideal lattice $\sigma(\mathcal{O}_{\mathbf{K}})$ should be small in a certain sense. Indeed the coefficients of T_F, that are the symmetric polynomials in the coordinates of the vector, are expected to be small. However, we need to replace the notion of smallness by something more adapted than the L_2-norm of the vector.

This is precisely the idea behind our algorithm: to consider all small enough vectors in the lattice until we find one that defines a minimal polynomial and can prove that this polynomial is indeed of minimal height. In order to do that, let us first focus on the target solution T_F and the corresponding vector $\sigma(\theta_F) = [\sigma_1(\theta_F),\dots,\sigma_n(\theta_F)]$. Remark that when all entries of T_F have roughly the same size, its L_2-norm is also small. In that case, the algorithm of Cohen and Diaz y Diaz succeeds in finding T_F. However, when the sizes of the entries are unbalanced, success is no longer guaranteed. For a more precise analysis, let us introduce a parameter $c > 1$ whose value is determined later to minimize algorithmic complexity[2]. Now, consider the vector $\big[\log_c|\sigma_1(\theta_F)|,\dots,\log_c|\sigma_n(\theta_F)|\big]$ that describes the relative sizes of the coordinates of $\sigma(\theta_F)$. Then, round each value up to the smallest possible non-negative integer, this gives a vector $b^F = \big[b_1^F,\dots,b_n^F\big]$. Since T_F is monic, by definition, the Mahler measure $M(T_F)$ is the product of the values $\max(1,|\sigma_i(\theta_F)|)$ and we can write the following inequality between $M(T_F)$ and the vector b^F:

$$c^{\sum b_j^F - n} \quad \leq \quad M(T_F) \quad \leq \quad c^{\sum b_j^F}.$$

In particular, this guarantees thanks to Equation (1.7) that

$$\sum_{j=1}^{n} b_j^F \quad \leq \quad \log_c M(T_F) + n \quad \leq \quad \log_c H(T_F) + \frac{1}{2}\log_c(n+1) + n. \tag{3.4}$$

[1] We recall that we consider the smallest-height defining polynomial among the monic polynomials

[2] It is done in Section 3.3.3 and it turns out that the optimal choice is $c = \exp(1)$.

Given the vector b^F of weights b_i^F, we can introduce a *weighted* version of the lattice corresponding to $\mathcal{O}_{\mathbf{K}}$ in $\mathbf{R}^{r_1} \times \mathbf{C}^{r_2}$. The weighted lattice is generated by the vectors

$$\widetilde{\Omega_i} = \left[\frac{\sigma_1(\omega_i)}{c^{b_1^F}}, \dots, \frac{\sigma_n(\omega_i)}{c^{b_n^F}} \right].$$

In this new lattice, θ_F is associated to the vector

$$\tilde{\sigma}(\theta_F) = \left[\frac{\sigma_1(\theta_F)}{c^{b_1^F}}, \dots, \frac{\sigma_n(\theta_F)}{c^{b_n^F}} \right].$$

Note that, by construction, each coordinate of $\tilde{\sigma}(\theta_F)$ is absolutely bounded by 1 so that its L_2-norm is bounded by $\sqrt{n}$.

As in Section 2.4.1, we could work with the Gram matrix of the lattice and, for instance, use the implementation provided by [EJ17]. However, as an alternative and because it eases the explanation, we prefer to describe the enumeration process by using real vectors. Thus we replace the lattice in $\mathbf{R}^{r_1} \times \mathbf{C}^{r_2}$ by the related lattice in $\mathbf{R}^n$, as explained in Section 1.2.2. A third option would be to work directly with the complex lattice as it is described in [GLM09]. First, remark that since $\sigma_{r_1+i} = \overline{\sigma_{r_1+i+r_2}}$, we necessarily have $b^F_{r_1+i} = b^F_{r_1+i+r_2}$. As a consequence, only the first $r = r_1 + r_2$ coefficients of the weight vector b^F are needed to describe the weighted lattice, which is generated by

$$\widetilde{\Omega_i}^{\mathbf{R}} = \left[\frac{\sigma_1(\omega_i)}{c^{b_1^F}}, \dots, \frac{\sigma_{r_1}(\omega_i)}{c^{b_{r_1}^F}}, \frac{\sqrt{2}\,\Re\left(\sigma_{r_1+1}(\omega_i)\right)}{c^{b_{r_1+1}^F}}, \frac{\sqrt{2}\,\Im\left(\sigma_{r_1+1}(\omega_i)\right)}{c^{b_{r_1+1}^F}}, \dots, \frac{\sqrt{2}\,\Im\left(\sigma_{r_1+r_2}(\omega_i)\right)}{c^{b_{r_1+r_2}^F}} \right].$$

In the lattice generated by the $\widetilde{\Omega_i}$, the vector corresponding to the integer θ_F has all its entries of norm at most 1. Thus its L_2-norm is bounded by $\sqrt{n}$. Due to the equality of the L_2-norm (see Remark 1.2.7), this bound also holds in the lattice generated by the $\widetilde{\Omega_i}^{\mathbf{R}}$.

As a consequence, our algorithm simply aims at enumerating all the possible weighted lattices — *i.e.*, all the possible vectors of weights — and all vectors of L_2-norm at most $\sqrt{n}$ in these lattices. However, doing that directly would be sub-optimal, since for any integer θ in $\mathcal{O}_{\mathbf{K}}$, we would also consider many shifted copies $\theta + j$ with $j \in \mathbf{Z}$. To avoid that, we work instead with a lattice projected orthogonally with respect to the vector $\widetilde{\Omega_1}^{\mathbf{R}}$ to find a candidate θ. Then, finding the best $\theta + j$ takes polynomial time in n. The resulting algorithm is described as Algorithm 2 and how finding j and precision issues are discussed later in this chapter.

If the *a priori* bound B_F is correct, we know that θ_F belongs to one of the weighted lattices that we have encountered. Thus, the last memorized polynomial in Algorithm 2 is — one of the possible — T_F. Otherwise, if the input bound does not hold, the algorithm asserts this fact, as there is no polynomial stored.

Algorithm 2 Search for a minimal-height defining polynomial.

Input: An integral basis $\omega_1, \ldots, \omega_n$ of $\mathcal{O}_{\mathbf{K}}$ and an *a priori* bound B_F on $\log_c H(T_F)$.

Output: A polynomial T_F whose height is minimal among all the defining polynomials.

1: Fix $k = 0$, let Bound $= \lfloor B_F + \frac{1}{2}\log_c(n+1) \rfloor + n$

2: **for** $(b_1, \ldots, b_n) \in \mathbf{N}^n$ such that $b_{r_1+i} = b_{r_1+r_2+i}$ and $\sum_{j=1}^{n} b_j = k$ **do**

3: Construct the weighted lattice generated by the vectors

$$\widetilde{\Omega_i}^{\mathbf{R}} = \left[\frac{\sigma_1(\omega_i)}{c^{b_1}}, \ldots, \frac{\sigma_{r_1}(\omega_i)}{c^{b_{r_1}}}, \frac{\sqrt{2}\,\Re\left(\sigma_{r_1+1}(\omega_i)\right)}{c^{b_{r_1+1}}}, \ldots, \frac{\sqrt{2}\,\Im\left(\sigma_{r_1+r_2}(\omega_i)\right)}{c^{b_{r_1+r_2}}} \right]$$

(since the lattice has real entries, algorithmically we replace it by a sufficiently precise fixed point approximation and scale it up to an integer lattice)

4: Project the vectors $\widetilde{\Omega_i}^{\mathbf{R}}$ for $i \in \{2, \ldots, n\}$ orthogonally with respect to $\widetilde{\Omega_1}^{\mathbf{R}}$

5: Enumerate all vectors of L_2-norm at most[3] $1.001\sqrt{n}$ in this projected lattice

6: **for** each short vector $v^{\perp}$ **do**

7: Lift $v^{\perp}$ to the shortest possible vector v in the full lattice (this can be done efficiently by using Gauss' algorithm on the two-dimensional lattice spanned by $\widetilde{\Omega_1}^{\mathbf{R}}$ and an arbitrary lift of $v^{\perp}$)

8: Reconstruct the corresponding polynomial

$$T_v(X) = \prod_{j=1}^{r_1} (X - v_j) \cdot \prod_{j=1}^{r_2} \left((X - v_{r_1+2j-1} - i\, v_{r_1+2j}) \cdot (X - v_{r_1+2j-1} + i\, v_{r_1+2j}) \right)$$

9: **if** T_v is irreducible **then**

10: Find the integer j that minimizes the height of $T_v(X + j)$

11: Store $T_v(X + j)$ if it has the smallest height encountered until this point

12: When storing $T_v(X + j)$,

update Bound to $\lfloor \log_c H(T_v(X + j)) + \frac{1}{2}\log_c(n+1) \rfloor + n$

13: **end if**

14: **end for**

15: **end for**

16: Increment k by 1

17: **if** $k \leq$ Bound **then goto** 2

18: **else** *Stop*

19: **end if**

[3] With an exact representation of the lattice, it would suffice to enumerate vectors of norm below $\sqrt{n}$. The 1.001 factor comes from the fact that we are dealing with an approximation of the lattice. See the paragraph on precision.

Finding j. In the algorithm, we need to find the integer j that minimizes the height of $T_v(X+j)$. This can be done in polynomial time. Indeed, each coefficient of $T_v(X+j)$ (viewed as a polynomial in X) is a polynomial in j. We thus want to minimize (over the integers) the maximum of the absolute values of n polynomials — since $T_v(X+j)$ is monic, we do not need to include the coefficient in front of X^n in the minimization. This can be restated as minimizing the maximum of $2n$ polynomials: the coefficients and their opposites. This function is a continuous positive piecewise-polynomial real function. The transition from one piece to the next necessarily occurs at a root of the difference between two out of the $2n$ polynomials. Thus, there are at most $O(n^3)$ pieces. In each piece, it is easy to minimize over the reals and then over the integers by considering the two integers closest to this real minimum — ignoring the integer values lying outside of the current piece. As a consequence, the best value for j can be found in polynomial time.

Precision. At the beginning of the algorithm, the lattice generated by the vectors $\widetilde{\Omega_i}^{\mathbf{R}}$ is replaced by a "sufficiently" precise fixed point approximation and then scaled up to an integer lattice. In order to make the description of the algorithm complete, we need to specify the corresponding precision. The key point is that the precision should be high enough to ensure that any short vector of the exact lattice corresponds to a short vector of the approximate lattice. Indeed, we are essentially enumerating all vectors of length bounded by $\sqrt{n}$ and we want to make sure that none of those can be overlooked. Let v be a vector of norm at most $\sqrt{n}$ of the exact lattice. We can write

$$v = \sum_{i=1}^{n} v_i \widetilde{\Omega_i}^{\mathbf{R}} \quad \text{with integer coeffients } v_i.$$

In order to make sure that the vector of the approximate lattice is also short, *i.e.,* shorter than $1.001\sqrt{n}$, the precision should be good enough for the sum of products of the coefficients v_i by the approximation error to be smaller than a constant. In particular, it is enough to require that the maximum of the v_i times the approximation error is smaller than $\frac{1}{1000\,n}$.

As a consequence, in order to bound the necessary precision, it suffices to upper bound the coefficients v_i. A classical technique to obtain such a bound is to start from a lower bound on the orthogonalized vectors of the initial basis and remark that the coefficients v_i are upper bounded in absolute value by $\sqrt{n}$ times the inverse of this bound. To obtain a lower bound on a given $\|b_i^*\|$, we can divide the determinant of the lattice by the norms of all the vectors but the corresponding b_i. The determinant of the scaled lattice at iteration k of the algorithm is $\sqrt{|\Delta_{\mathbf{K}}|}\,c^{-k}$. Thanks to [Poh93, Chapter V - Lemma 1.1], we know that, in the basis of the ring of integers of $\mathbf{K}$ computed from the input polynomial T_{in}, the vector ω_i is a polynomial of degree $i-1$ in the roots of T. Moreover, the coefficients of this polynomial are

in the interval $[-1,1]$. As a consequence, the norm of $\sigma(\omega_i)$ can be upper bounded by $i\sqrt{n}Z$ where Z denotes the largest complex modulus among the roots of T. In particular, $Z \leq M(T)$.

Putting all this together, we can upper bound the coefficients v_i at iteration k by

$$\frac{c^k M(T)^{n^2} n^{1.5n}}{|\Delta_{\mathbf{K}}|}.$$

Thus, it suffices to work with a precision $k\log_2(c) + n^2 \log_2 M(T) + 1.5n\log_2(n) + \log_2(1000n)$, which is polynomial in the size of the input polynomial. This precision is also sufficient for performing the reconstruction of T_v from the complex embeddings, rounding each coefficient to the nearest integer (see [Bel04, Section 3.2] for more details).

From a practical perspective, the implementation of Espitau and Joux can be used. Using *interval arithmetic*, they are able to perform lattice reduction — and so enumeration of small vectors — and certify that the approximated output result is an exact solution. In addition, as we mentioned, it works directly with the Gram matrix as input. Thanks to our analysis, we know that the required precision is reasonable, so that we can make use of their algorithm.

Pseudo-canonical polynomial. For some applications, such as the construction of tables of number fields, it is useful to have a canonical polynomial that represents a given number field. In that case, it is not difficult to add in the above algorithm an arbitrary criterion such as lexicographic order on the coefficients to single out one of the possible minimal-height polynomials. Indeed, all of them are encountered during the enumeration.

3.3 Complexity analysis

Now we have described our algorithm, in order to understand its algorithmic complexity, we need to count the number of lattices involved and to study the cost of short vector enumeration in each lattice. We restrict ourselves to primitive number fields for this analysis. For imprimitive fields, the algorithm still works but its complexity may be higher due the possibility of encountering a much larger number of algebraic integers that generate subfields during the short vector enumerations. We leave as an open problem the adaptation of the techniques used in `PARI/GP` [PARI] for the algorithm of Cohen and Diaz y Diaz with imprimitive fields to our method. In our complexity analysis, we ignore the cost of computing an integral basis for the maximal order of the field $\mathbf{K}$, because it is already required for the class group computation itself. According to Section 1.2.4, the complexity of this step is dominated by the factorization of the discriminant of the input polynomial. Thanks to the NFS, this can be done in time $L\left(\frac{1}{3}\right)$ and depending on T_{in}, it may be the most costly part of the complete computation.

3.3.1 Number of lattices

When the input bound B_F is incorrect, the number of lattices that are explored is a function of B_F. When B_F is correct, thanks to the updating of the bound when a new polynomial is found, the number of lattices depends on the highest value of k that is reached and this value is at most $\log_c H(T_F) + \frac{1}{2}\log_c(n+1) + n$ according to Equation (3.4). Thus, the exact runtime depends on the quality of the output: it is faster to find polynomials with a smaller height.

Let us consider each iteration of the outer loop: for iteration k, the number of lattices is simply the number of vectors $(b_1, \ldots, b_n) \in \mathbf{N}^n$ that satisfy the two constraints $b_{r_1+i} = b_{r_1+r_2+i}$ and $\sum_{j=1}^{n} b_j = k$. To obtain an upper bound, we can forget the first constraint and just count the number of vectors such that $\sum b_j = k$. It is a classical combinatorial result that this number is $\binom{k+n-1}{n-1}$. Hence, the total number of lattices considered by the algorithm is

$$\sum_{k=0}^{k_F} \binom{k+n-1}{n-1} = \binom{k_F+n}{n} \leq \frac{(k_F+n)^n}{n!}, \tag{3.5}$$

where k_F is the value of k in the last iteration, *i.e.,*

$$k_F = \left\lfloor \min\left(B_F, \log_c H(T_F)\right) + \frac{1}{2}\log_c(n+1) \right\rfloor + n.$$

Bound on k_F and classes $\mathcal{D}_{n_0,d_0,\alpha,\gamma}$. Given as input a number field that we know to belong to a class $\mathcal{D}_{n_0,d_0,\alpha,\gamma}$, we can simply set the value of B_F given to the algorithm to

$$B_F = \left\lfloor \frac{d_0}{\log c} (\log|\Delta_{\mathbf{K}}|)^{\gamma} (\log\log|\Delta_{\mathbf{K}}|)^{1-\gamma} \right\rfloor.$$

Moreover, since we are limiting ourselves to the cases $\alpha \leq \frac{1}{2} \leq \gamma$, then n is negligible compared to B_F and $k_F + n = B_F(1+o(1))$.

We conclude that the number of lattices in the enumeration is upper bounded by $\frac{(B_F(1+o(1)))^n}{n!}$ and so by $L_{|\Delta_{\mathbf{K}}|}(\alpha, \gamma\, n_0 - \frac{\alpha}{n_0})$.

3.3.2 Cost of each enumeration

The second analysis treats the enumeration phase for each weighted lattice. We recall that we are interested in all vectors whose Euclidean norm is below $\sqrt{n}$. A slight modification of Kannan's SVP algorithm leads to a satisfactory enumeration approach, the complete analysis of which can be found in [HS07]. Note that the resulting enumeration cost includes the HKZ reduction of the basis.

Proposition 3.3.1. *Enumerating all vectors of norm below $\sqrt{n}$ in one of our weighted lattices can be done in at most* $\mathrm{Poly}\left(\log|\Delta_{\mathbf{K}}|\right) \cdot n^{\frac{n}{2e}+o(n)}$ *binary operations.*

Proof. This is mostly a consequence of the results of [HS07], which gives a two-step strategy for enumerating the short vectors of a lattice. The first step is to compute an HKZ-reduced basis of the lattice and the second to enumerate the short vectors using this HKZ-reduced basis. Thanks to the high quality of the reduced basis, the enumeration process is greatly speeded up.

The complexity of the enumeration process is analyzed in [HS07, Section 4.1]. Assuming that we have a n-dimensional lattice given by a basis $(b_1,\ldots,b_n)$ with Gram-Schmidt orthogonalization denoted by $(b_1^*,\ldots,b_n^*)$ and that we wish to enumerate all vectors of L_2-norm at most A. Then the complexity expressed in number of arithmetic operations is

$$2^{O(n)} \cdot \max_{I\subset[1\ldots n]} \left(\frac{A^{|I|}}{\sqrt{n}^{|I|} \prod_{i\in I} \|b_i^*\|} \right).$$

Moreover, [HS07, Theorem 3] states that for an HKZ-reduced basis, for all subsets I of $[1\ldots n]$, we have

$$\frac{\|b_1\|^{|I|}}{\prod_{i\in I}\|b_i^*\|} \quad \leq \quad \sqrt{n}^{|I|+\frac{n}{e}}.$$

As a consequence, in terms of arithmetic operations, enumerating all vectors of L_2-norm at most $\lambda\|b_1\|$ costs at most

$$2^{O(n)} \cdot \sqrt{n}^{\frac{n}{e}} \cdot \lambda^n.$$

The preliminary step consisting in computing an HKZ-reduction of the lattice before doing the enumeration does not dominate the complexity of the enumeration itself (see [HS07, Theorem 2]).

In [HS07], the complexity is expressed in terms of bit operations for the case of an integer lattice, given a bound on the larger integers appearing in the lattice basis. In our case, as already explained after Algorithm 2, we work with an integer lattice built from approximations, with entry sizes polynomial in the size of the input polynomial.

Finally, to apply the result to our case, we need to note that whenever we are enumerating in the lattice corresponding to a weight vector $[b_1,\ldots,b_n]$, we know that the weight vector $[\max(b_1-1,0),\ldots,\max(b_n-1,0)]$ cannot have led to a successful short vector during the enumeration earlier in the algorithm. Thus, due to the primitivity of the field, there is no vector of L_2-norm smaller than $\sqrt{n}$ in this lattice. As a consequence, there is no non-zero vector of L_2-norm smaller than $\frac{\sqrt{n}}{c}$ in the current lattice, associated to $[b_1,\ldots,b_n]$. Therefore we are

performing enumeration with a ratio factor λ that is at most c. We conclude that the cost of enumeration in bit operations is upper bounded by

$$\mathrm{Poly}\left(\log|\Delta_{\mathbf{K}}|\right)\cdot 2^{O(n)}\cdot n^{\frac{n}{2e}}\cdot c^n \quad \leq \quad \mathrm{Poly}\left(\log|\Delta_{\mathbf{K}}|\right)\cdot n^{\frac{n}{2e}+o(n)}. \tag{3.6}$$

Note that for the first lattice in the enumeration process — *i.e.*, $\sigma(\mathcal{O}_{\mathbf{K}})$, the one used by Cohen and Diaz y Diaz in their algorithm — it is also directly known that it contains no vectors of L_2-norm smaller than $\sqrt{n}$. □

We remark that $n^{\frac{n}{2e}+o(n)} = L_{|\Delta_{\mathbf{K}}|}\left(\alpha, \frac{\alpha\, n_0}{2e}\right)$. By multiplying with the number of lattices involved, we find that the final complexity can be expressed as

$$L_{|\Delta_{\mathbf{K}}|}\left(\alpha, \gamma\, n_0 - \frac{\alpha}{n_0} + \frac{\alpha\, n_0}{2e}\right),$$

which becomes arbitrarily close to $L_{|\Delta_{\mathbf{K}}|}\left(\alpha, \gamma - \frac{2e-1}{2e}\,\alpha\right)$ when n_0 is taken close to 1.

3.3.3 Optimal choice for c

Once the complexity analysis is done, it remains to study carefully the dependence on c in order to minimize the constant term. Assuming that the output polynomial defines a number field $\mathbf{K}$ in $\mathcal{D}_{n_0,d_0,\alpha,\gamma}$ such that $\gamma \geq \alpha$ — namely the case of application to class groups — then the global complexity we found is bounded by (see Equation (3.6))

$$\frac{\left(\frac{d_0}{\log c}(\log|\Delta_{\mathbf{K}}|)^{\gamma}(\log\log|\Delta_{\mathbf{K}}|)^{1-\gamma}(1+o(1))\right)^n}{n!}\cdot \mathrm{Poly}\left(\log|\Delta_{\mathbf{K}}|\right)\cdot 2^{O(n)}\cdot n^{\frac{n}{2e}}\cdot c^n.$$

As we want to find the best c, we only look at the dependence on c, which can be approximated as

$$\left(\frac{1}{\log c}\right)^n \cdot c^n = \left(\frac{c}{\log c}\right)^n,$$

and whose minimum is achieved by fixing c at $\exp(1)$.

3.4 Application to class group computation

Let us get back to class group computation. We recall that Biasse and Fieker find a way to reduce the complexity of their algorithm for certain classes of — orders in — number fields. Given a number field $\mathbf{K} \in \mathcal{C}_{n_0,d_0,\alpha}$ — defined by a polynomial T satisfying Equation (3.1) — they are able to compute the class group and a compact representation of a fundamental system of

units of this order in time $L_{|\Delta_{\mathbf{K}}|}(a,c)$ for $\frac{1}{3} \leq a < \frac{1}{2}$ depending on the parameters of the class and $c > 0$.

We now want to investigate the theoretical impact of using a minimal-height defining polynomial for class group computations. Note that the existence of such a polynomial ensures that $\mathbf{K}$ lies in a certain class $\mathcal{D}_{n_0,d_0,\alpha,\gamma}$ with $\gamma \in [1-\alpha, 1]$. If γ reaches the lower bound $\gamma = 1-\alpha$, then $\mathbf{K} \in \mathcal{D}_{n_0,d_0,\alpha,1-\alpha}$ and we can apply the conditional improvement[4] of [BF14] to this field. Our idea is then to add our algorithm as a precalculation to theirs in order to find the parameters required to fit $\mathbf{K}$ in such a class, if possible.

Doing this enables to extend the applicability of the algorithm of Biasse and Fieker to every number field in a class $\mathcal{D}_{n_0,d_0,\alpha,1-\alpha}$, even if the number field is not initially given by a small-height defining polynomial. It remains to check that the cost of this precalculation does not outweigh the global complexity of the rest of the class group computation. We know from Section 3.3 that our algorithm runs in time $L_{|\Delta_{\mathbf{K}}|}(\alpha)$ in that case. This never dominates the cost of their class group algorithm, thus we can state the following extended result.

Theorem 3.4.1. *Under ERH and smoothness heuristics, for every number field $\mathbf{K} \in \mathcal{D}_{n_0,d_0,\alpha,1-\alpha}$, there exists an $L_{|\Delta_{\mathbf{K}}|}(a,c)$ algorithm for class group and unit group computation for some $c > 0$ and a satisfying $a \geq \max\left(\alpha, \frac{1-\alpha}{2}\right)$.*

Furthermore, our new classes $\mathcal{D}$ built from four parameters allow us to more widely generalize the work of Biasse and Fieker to all number fields having $\alpha \leq \frac{1}{2}$:

Theorem 3.4.2. *Under ERH and smoothness heuristics, for every number field $\mathbf{K} \in \mathcal{D}_{n_0,d_0,\alpha,\gamma}$, there exists an $L_{|\Delta_{\mathbf{K}}|}(a,c)$ algorithm for class group and unit group computation for some $c > 0$ and $a = \max\left(\alpha, \frac{\gamma}{2}\right)$.*

Proof. This is only a generalization of the work of Biasse and Fieker and we make use of the same heuristics. Their final result mainly relies on [BF14, Theorem 3.1]. In fact we need to adapt the norm bounds appearing in the proof of this theorem to our more general classes. More precisely, we find that:

For all number fields $\mathbf{K}$ in $\mathcal{D}_{n_0,d_0,\alpha,\gamma}$, we can find a B-smooth ideal equivalent to a $|\Delta_{\mathbf{K}}|$-smooth ideal $\mathfrak{a} \subseteq \mathcal{O}_{\mathbf{K}}$ with a decomposition in degree 1 prime ideals in time $L_{|\Delta_{\mathbf{K}}|}(b,\mu)$ for some $\mu \geq 0$ where $B = Ł_{|\Delta_{\mathbf{K}}|}(a,\rho)$ for some $\rho \geq 0$, provided that a and b satisfy

$$(i)\ b \leq \gamma \leq a+b; \qquad (ii)\ \alpha+\gamma \leq a+2b; \qquad (iii)\ \alpha+\gamma \leq 2a+b; \qquad (iv)\ \alpha \leq b.$$

[4] More precisely the adaptation of this improvement to the classes $\mathcal{C}'$.

The proof is exactly the same, except that we modify the exponent appearing in the definition of k. The value $1-\beta-\frac{\tau}{2}$ in their proof then becomes $\alpha+\gamma-\beta-\frac{\tau}{2}$ in our generalization and the remaining part follows as in [BF14]. □

We thus conclude from this theoretical analysis that our algorithm for reducing number field defining polynomials widens the set of number fields whose class group and unit group can be computed with runtime $L_{|\Delta_K|}(a)$, $\frac{1}{3} \leq a < \frac{1}{2}$. The experimental results below illustrate that it also offers a good behavior on practical examples.

3.4.1 Experimental results

We have implemented a prototype of our algorithm under `MAGMA` 2.21.6[5] [Magma]. The precision and the parameter c are fixed by the user. It slightly differs from the description in Section 3.2 since our code enumerates all the elements of norm below $\sqrt{n}$ in each weighted lattice, instead of doing the minimization of the height of $T_v(X+j)$. Indeed, for the practical examples we have considered, it is faster to proceed in that way. Furthermore, when a polynomial T (better than the input polynomial) is found, our code offers two options. The first is to stop immediately, since it appears that in practice the first polynomial to be found usually has minimal height. Despite the fact that this *early abort* does not certify the minimality, it is a good practical compromise for class group computations. Another approach when a first polynomial is found is to directly increment k to the value $\lfloor \log_c H(T) + \frac{1}{2}\log_c(n+1) \rfloor + n$. This skips intermediate computations which are often useless and directly goes to checking that the polynomial T has minimal height.

In our implementation, we do not fix the parameter c to $\exp(1)$ as in the asymptotic analysis. Instead, we remark that using a large c allows for smaller values of k and makes the code run faster for finding a first candidate with small height. However, to find a minimal-height polynomial, smaller values of c are better. This can be seen on the examples given in Table 1.

Example 3.4.3. As a first benchmark, we consider the polynomial $t^{12}+4t^{11}-17t^{10}-68t^9+108t^8+416t^7-314t^6-1129t^5+358t^4+1353t^3-36t^2-540t-72$ given as an example in [Poh93, Section V.3]. This polynomial is obtained by using the algorithm of Cohen and Diaz y Diaz to a polynomial with huge coefficients; it defines a suborder of index 670150656 of the ring of integers. More precisely, the vector corresponding to the above polynomial appears as the second vector in the reduced basis after LLL reduction. The fourth vector of the same basis yields a better polynomial with smaller height (505) and smaller index (439826112).

[5] Our prototype is prior to the results of Espitau and Joux [EJ17]. It would be interesting to refine our code using theirs.

With our algorithm , we find an even better polynomial:

$$t^{12} - 14\,t^{11} + 25\,t^{10} + 62\,t^9 - 155\,t^8 - 50\,t^7 + 263\,t^6 - 50\,t^5 - 155\,t^4 + 62\,t^3 + 25\,t^2 - 14\,t + 1,$$

whose associated order has index 419904 — and whose height is 263. In addition, this polynomial is palindromic and, thus, reveals an underlying symmetry of the corresponding number field which was not apparent from the other defining polynomials.

Example 3.4.4. To illustrate practically what changes in class group computation, we choose an example having smaller degree and larger coefficients:

$$x^5 - 5843635x^4 + 931633x^2 + 6577x - 8570.$$

Our implementation of the reduction algorithm certifies that this polynomial has minimal height.

If we give it to MAGMA V2.21.6 in order to compute the class group of the associated number field, it first "reduces" the polynomial as

$$x^5 - 2x^4 - 8001397580x^3 - 31542753393650x^2 + 3636653302451131875x + 4818547529425280067500$$

and then finds the class group — assuming ERH — in about 285 seconds, on the laptop we used for all our experiments. More precisely, according to the output of the verbose mode, it seems that MAGMA derives the relations by sieving on different polynomials. Those polynomials are chosen as minimal polynomials of algebraic integers of the form $\frac{\theta+b}{c}$, where θ is the second element of the LLL-reduced integral basis of $\mathcal{O}_{\mathbf{K}}$. We have reconstructed the "reduced" polynomial from this information: it is the minimal polynomial of θ. MAGMA uses 2306 different sieving polynomials: 663 lead to 0 relations, 748 to 1 relation, 498 to 2 relations and 397 to at least 3 relations. None of them produce more than 8 relations.

In MAGMA V2.19.10, class group computation is not as optimized as in V2.21, but there is no reduction algorithm: MAGMA directly works with the input polynomial. Thus, we can compare the efficiency of using either of the two polynomials:

- with $x^5 - 5843635x^4 + 931633x^2 + 6577x - 8570$, it takes about 140 seconds,
- with the "reduced" one, it takes about 3240 seconds.

We see that using the old version with a minimal-height polynomial is even faster than the new version, despite the huge optimizations in the class group computation code. For a more detailed comparison, using the best polynomial as input, the old version only sieves on 58 different polynomials and the first one already leads to 784 relations. Consequently, we doubt of the efficiency of the reduction step added in the new version.

Note that our implementation in MAGMA finds this minimal-height polynomial from the "reduced" one in less than 1.5 second. It also certifies that it is indeed of smallest height. Thus, in this practical case, the reduction algorithm takes negligible time and using it as a precomputation can greatly speed up the class group computation.

Example 3.4.5. Finally, we run our implementation on sets of number fields stored in the online Class Group Database [CGD]. For each degree $n \in \{3,4,5,7\}$, we pick 100 number fields having discriminant about b bits and we try to reduce the polynomial T_0 given in the table. For each degree, we consider two different sizes of discriminants in order to observe how the algorithm behaves as the discriminant grows.

To illustrate the improvement compared to the previously tabulated polynomial, we compute the ratio $r = \frac{\log H(T_F)}{\log H(T_0)}$, where T_F denotes the output polynomial.

n	$\log_2 \lvert\Delta_{\mathbf{K}}\rvert$	Early abort					Proven minimal height				
		c	avg. r	min r	max r	avg. time	c	avg. r	min r	max r	avg. time
3	20	10	0.652	0.435	0.860	7 ms	2	0.651	0.435	0.860	49 ms
	40	10	0.635	0.381	0.741	17 ms	3	0.634	0.381	0.736	131 ms
4	25	10	0.607	0.393	0.868	18 ms	2	0.604	0.393	0.868	1.0 s
	40	10	0.547	0.446	0.806	108 ms	3	0.544	0.446	0.806	6.6 s
5	50	10	0.724	0.567	0.955	109 ms	3	0.715	0.567	0.955	4.8 s
	80	10	0.698	0.555	0.852	820 ms	4	0.695	0.555	0.852	14.7 s
7	80	20	0.712	0.519	0.970	1.7 s	4	0.701	0.519	0.970	749 s
	100	20	0.700	0.537	0.843	6.3 s	–	–	–	–	–

Table 1: Gain in the size of the height after reduction, $r = \frac{\log H(T_F)}{\log H(T_0)}$

As a conclusion, for every number field in the database, there exists a defining polynomial with smaller height, sometimes as small as the square root of the height of the input polynomial. In addition, we can observe the good behavior of the early-abort version since the values obtained in that way closely coincide with the minimal values.

Chapter 4

Refinements for complexities appearing in the literature for the general case

Contents

Before we consider the consequences of having small defining polynomials, which is done in the next chapter, we take care of the general case, when no good polynomials are known and the extension degree is too large to find one. As recalled in Section 2.3.3, the current best algorithm for the general case is the algorithm presented by Biasse and Fieker in [BF14]. Their work is already mostly described in Chapter 2 so we only present the relation collection here. After describing the way they do it, we give a simplified version of the algorithm and a detailed analysis of its complexity. We then provide a more accurate and improved complexity, using the classes $\mathcal{D}$ we have introduced in Chapter 3. Finally we present an improved version of the collection, based on the Cheon's trick detailed in Section 1.1.3, in order to further reduce the complexity for large degree number fields.

4.1 The classification defined by classes $\mathcal{D}$ is sufficient

For the discrete logarithm problem in finite fields, all the fields are classified according to the relative size of their characteristic — small, medium or large. Our purpose is to derive a similar classification for the number fields. For finite fields, the cardinality Q is completely determined by the characteristic p and the extension degree n, according to the equation $Q = p^n$. For number fields, the extension degree remains, but the characteristic is replaced by the size of the defining polynomial, represented by its height $H(T)$. Unfortunately, number fields do not provide any equality similar to $Q = p^n$ for finite fields, but only the inequality of Equation (1.8):

$$|\Delta_{\mathbf{K}}| \leq n^{2n} H(T)^{2n-2}.$$

Therefore, we choose the extension degree as the main parameter of our classification. The Minkowski's bound (Equation (2.3)) induces that $n = O(\log|\Delta_{\mathbf{K}}|)$, because every non-zero integral ideal has a norm in $\mathbf{N}^*$. Thus we want to express n in terms of $\log|\Delta_{\mathbf{K}}|$. Fortunately, this choice is a perfect match with the classes $\mathcal{D}$ introduced in Chapter 3.

Definition 4.1.1. Let $n_0 > 1$ be a real parameter arbitrarily close to 1, $d_0 > 0$, $\alpha \in [0,1]$ and $\gamma \geq 1-\alpha$. The class $\mathcal{D}_{n_0,d_0,\alpha,\gamma}$ is defined as the set of all number fields $\mathbf{K}$ of discriminant $\Delta_{\mathbf{K}}$ that admit a monic defining polynomial $T \in \mathbf{Z}[X]$ of degree n that satisfies Equation (3.3):

$$\begin{aligned} \frac{1}{n_0}\left(\frac{\log|\Delta_{\mathbf{K}}|}{\log\log|\Delta_{\mathbf{K}}|}\right)^{\alpha} \quad \leq \quad n \quad \leq \quad n_0\left(\frac{\log|\Delta_{\mathbf{K}}|}{\log\log|\Delta_{\mathbf{K}}|}\right)^{\alpha} \qquad \text{and} \\ d = \log H(T) \quad \leq \quad d_0(\log|\Delta_{\mathbf{K}}|)^{\gamma}(\log\log|\Delta_{\mathbf{K}}|)^{1-\gamma}. \end{aligned}$$

We recall that the factor $\log\log|\Delta_{\mathbf{K}}|$ is introduced to simplify the complexity analysis, while the condition $\gamma \geq 1-\alpha$ is a direct consequence of Equation (1.8). We emphasize that

the extension degree carries more information than the size of the coefficients of a defining polynomial — while giving the extension degree or the characteristic of a finite field carries the same information. Indeed, there exists an infinity of defining polynomials, and the quality of the smallest one depends on the number field: it is not known that we can always find one satisfying the lower bound $\gamma = 1 - \alpha$. That is why classifying number fields by their extension degree n — that is by $\alpha \in [0,1]$ — makes more sense. Then, for each α, there exists additional disparities according to γ, which is always greater than $1-\alpha$.

Thanks to the algorithm described in Chapter 3, we can restrict our study to the classes $\mathcal{D}$ with $\gamma \leq 1$ when α is in $\left[0, \frac{1}{2}\right]$. In these cases, Theorem 3.4.2 shows that we can compute the class group in time below $L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}\right)$. When $\alpha \geq \frac{1}{2}$, it is too costly to look for a small polynomial. We focus in this chapter on *large degree* number fields, the ones where $\alpha \geq \frac{1}{2}$.

At this point, it still remains to prove that considering classes $\mathcal{D}$ with $\alpha \in [0,1]$ suffices. At first sight, the Minkowski Theorem only results in $n = O(\log|\Delta_\mathbf{K}|)$ and implies that every number field belongs to a class $\mathcal{D}$ with $\alpha \leq 1+\varepsilon$ for an arbitrarily small $\varepsilon > 0$. However, a more accurate analysis leads to the following result:

Proposition 4.1.2. *Given $n_0 > 1$ and $\alpha > 1$, there does not exist an infinite family $(\mathbf{K}_i)_{i\geq 1}$ of number fields with discriminants $|\Delta_{\mathbf{K}_i}|$ and degrees n_i that satisfy*

$$\frac{1}{n_0}\left(\frac{\log|\Delta_{\mathbf{K}_i}|}{\log\log|\Delta_{\mathbf{K}_i}|}\right)^\alpha \quad \leq \quad n_i \quad \leq \quad n_0\left(\frac{\log|\Delta_{\mathbf{K}_i}|}{\log\log|\Delta_{\mathbf{K}_i}|}\right)^\alpha.$$

Proof. We proceed by contradiction. Let $(\mathbf{K}_i)_{i\geq 1}$ be an infinite family of number fields whose degrees n_i satisfy

$$\frac{1}{n_0}\left(\frac{\log|\Delta_{\mathbf{K}_i}|}{\log\log|\Delta_{\mathbf{K}_i}|}\right)^\alpha \leq n_i.$$

We provide an upper bound in the statement of the proposition as it is in the definition of classes $\mathcal{D}$. However, we only consider this inequality because it is the one that is problematic. The Minkowski's bound (Equation (2.3)), derived from the theorem states that for a field $\mathbf{K}$ of degree n,

$$\frac{n^n}{n!}\cdot\left(\frac{\pi}{4}\right)^{\frac{n}{2}} \leq \sqrt{|\Delta_\mathbf{K}|}. \tag{4.1}$$

Combining Equation (4.1) with the inequality $n! \leq e\, n^{n+\frac{1}{2}}\, e^{-n}$ derived from the Stirling formula [Moi30, Sti30], we obtain $n\left(2+\log\frac{\pi}{4}\right) \leq \log|\Delta_\mathbf{K}| + 2 + \log n$. Let A denote the constant $2+\log\frac{\pi}{4} > 1$. Then for all $i \geq 1$, we have

$$\frac{A}{n_0}\left(\frac{\log|\Delta_{\mathbf{K}_i}|}{\log\log|\Delta_{\mathbf{K}_i}|}\right)^\alpha \quad \leq \quad \log|\Delta_{\mathbf{K}_i}| + 2 + \log n_0 + \alpha\left(\log\log|\Delta_{\mathbf{K}_i}| - \log\log\log|\Delta_{\mathbf{K}_i}|\right)$$

$$\Longrightarrow \qquad 0 \;<\; \frac{A}{n_0} \;\leq\; \frac{\left(\log\log|\Delta_{\mathbf{K}_i}|\right)^{\alpha}}{\left(\log|\Delta_{\mathbf{K}_i}|\right)^{\alpha-1}} + \left(2+\log n_0+\alpha\log\log|\Delta_{\mathbf{K}_i}|\right)\cdot\left(\frac{\log\log|\Delta_{\mathbf{K}_i}|}{\log|\Delta_{\mathbf{K}_i}|}\right)^{\alpha}.$$

Finally, as the set of number fields having bounded discriminant is finite, it follows from our initial assumption that the family of discriminants $\left(|\Delta_{\mathbf{K}_i}|\right)_{i\geq 1}$ tends to infinity. But in that case, as α is chosen strictly greater than 1, the right hand side tends to 0, which leads to a contradiction. □

Example 4.1.3. To illustrate this proposition, we consider cyclotomic fields, which are known to be fields with small discriminants and large degrees.

For the l-th cyclotomic field $\mathbf{K}=\mathbf{Q}(\zeta_l)$, with $l=\prod p_i^{k_i}$ and denoting by φ the Euler totient function, the extension degree satisfies

$$[\mathbf{Q}(\zeta_l):\mathbf{Q}] = \varphi(l) = \prod\varphi\left(p_i^{k_i}\right) = \prod(p_i-1)p_i^{k_i-1},$$

and the discriminant is (see [Was97, Proposition 2.7])

$$|\Delta_{\mathbf{K}}| = \frac{l^{\varphi(l)}}{\prod p_i^{\varphi(l)/p_i-1}}.$$

Thus we obtain

$$\varphi(l) = \frac{\log|\Delta_{\mathbf{K}}|}{\log\log|\Delta_{\mathbf{K}}|}\cdot\frac{\sum(k_i-1)\log p_i+\log(p_i-1)}{\sum(k_i-\frac{1}{p_i-1})\log p_i}\big(1+o(1)\big), \tag{4.2}$$

and as $(k_i-1)\log p_i+\log(p_i-1)\approx(k_i-\frac{1}{p_i-1})\log p_i$ when p_i or k_i tends to infinity, we conclude that the ratio of the sums tends to 1 when l tends to infinity.

For instance, when $l=p$, the second factor in Equation (4.2) is $\frac{p-1}{p-2}\frac{\log(p-2)+\log\log p}{\log p}$, which tends to 1 as p goes to infinity, while for $l=p^k$ with p fixed and k tending to infinity, the second factor becomes $\frac{k}{k-\frac{1}{p-1}}(1+o(1))$.

Hence all cyclotomic fields asymptotically belong to a class $\mathcal{D}$ with $\alpha=1$. Finally, Proposition 4.1.2 leads to the following statement:

Corollary 4.1.4. *Asymptotically, the classes $\mathcal{D}_{n_0,d_0,\alpha,\gamma}$ with $\alpha\in[0,1]$ include all number fields.*

Note that despite Corollary 3.1.3, we do not specify the condition $\gamma\in[1-\alpha,1]$ in this result. Indeed when $\alpha\geq\frac{1}{2}$, finding the smallest height defining polynomial costs more than computing the class group. In these cases, it is preferable to work with the input polynomial. Another possibility is to perform only a partial reduction. More precisely, we may use the reduction algorithm described in [Coh93, Section 4.4] which consists in computing an LLL-reduced basis of the lattice of algebraic integers. Assuming that an integral basis is already

known, the runtime is polynomial in $\log|\Delta_{\mathbf{K}}|$. Eventually, for the rest of the chapter, we focus our study on classes $\mathcal{D}$ with $\alpha \in \left[\frac{1}{2}, 1\right]$ and $\gamma \geq 1-\alpha$. Indeed, although the algorithm works for $\alpha \leq \frac{1}{2}$, the complexity is larger than what we have obtained in Chapter 3.

4.2 The relation collection

The core idea is presented by Biasse in [Bia14b]: the generation of the relations based on BKZ-reductions of ideal lattices. The strategy is still the same as Buchmann's work: we reduce an ideal $\mathfrak{a}$, using lattice techniques, in order to find another ideal $\mathfrak{b}$ that belongs to the same class (see Section 2.3.2). While the algorithm of Buchmann looks for a shortest non-zero vector — whose runtime is polynomial in the size of the discriminant but exponential in the extension degree — the method of Biasse involves BKZ-reductions, that offer a trade-off between the time spent in the reduction and the approximation factor of the short vectors, as explained in Section 1.1.3. This leads to a subexponential algorithm that allows both the discriminant and the degree to tend to infinity. When combined with the linear algebra and regulator computation, it leads to the following theorem from [BF14]:

Theorem 4.2.1. *[BF14, Theorem 6.1] Under ERH and smoothness heuristics, the presented algorithm computes the class group structure together with compact representations of a fundamental system of units of a number field* $\mathbf{K}$ *of degree* n *and discriminant* $\Delta_{\mathbf{K}}$ *in time* $L_{|\Delta_{\mathbf{K}}|}(a)$ *for*

- $a = \frac{2}{3} + \varepsilon$ *for* $\varepsilon > 0$ *arbitrary small in the general case;*
- $a = \frac{1}{2}$ *when* $n \leq (\log|\Delta_{\mathbf{K}}|)^{3/4-\varepsilon}$ *for* $\varepsilon > 0$ *arbitrary small.*

Figure 1 presents the complexity of class group computations as a function of α, *i.e.,* the extension degree, prior to the improvements that are presented later in this chapter. It is based on the classification obtained in Section 4.1 and, for $\alpha \leq \frac{1}{2}$, on Chapter 3.

4.2.1 Description of the algorithm of Biasse and Fieker

In [Bia14b], and so in [BF14], the relation collection is derived from a reduction algorithm that given an ideal $\mathfrak{a}$ returns a smooth ideal $\mathfrak{b}$ that is in the same class as $\mathfrak{a}$. Then, applying this reduction to every ideal belonging to the factor base, we get the relations we are expecting.

We recall that the factor base $\mathcal{B} = \{\mathfrak{p}_1, \ldots, \mathfrak{p}_{|\mathcal{B}|}\}$ consists of all prime ideals of $\mathcal{O}_{\mathbf{K}}$ whose norm is below a bound $B = Ł_{|\Delta_{\mathbf{K}}|}(\beta, c_b)$, for $\beta \in [0,1]$ and $c_b > 0$. We also fix $\varepsilon > 0$ arbitrarily small and $\mathfrak{a}$ an ideal of $\mathcal{O}_{\mathbf{K}}$.

From the ideal $\mathfrak{a}$, Biasse and Fieker derive an ideal $\mathfrak{c}$, also in $\mathcal{O}_{\mathbf{K}}$, by $\mathfrak{c} = \mathcal{N}(\mathfrak{a}) \cdot \mathfrak{a}^{-1}$. This step consists of taking the inverse of $\mathfrak{a}$, and includes a norm multiplication to keep an integral ideal.

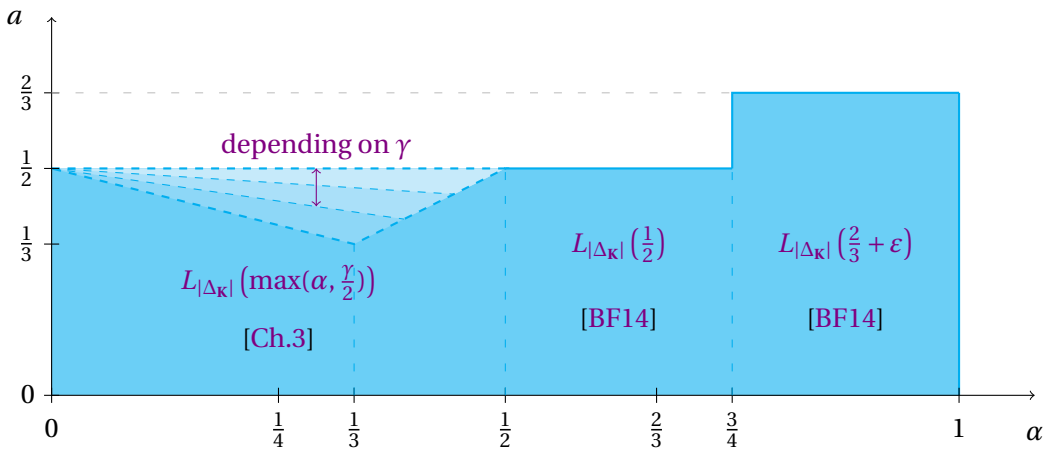

Figure 1: Complexity obtained by prior algorithms.

Then, similar to what Buchmann did, they choose an element $x \in \mathfrak{c}$, that is small in a certain sense, and define $\mathfrak{b}$ as the unique integral ideal that satisfies $\langle x \rangle = \mathfrak{c}\mathfrak{b}$. This $\mathfrak{b}$ is well-defined, as $x \in \mathfrak{c}$ implies $\langle x \rangle \subset \mathfrak{c}$. Finally, $\mathfrak{b}$ is in the same class as $\mathfrak{a}$, as $\mathfrak{b} = \langle x \rangle \mathfrak{c}^{-1} = \left\langle \frac{x}{\mathcal{N}(\mathfrak{a})} \right\rangle \mathfrak{a}$. The difference appears in the way to choose this *small* element $x \in \mathfrak{c}$: Biasse and Fieker replace shortest vector computations as used by Buchmann by BKZ-reductions.

Because of the results presented in Section 1.4.4 on smoothness of ideals in number fields, we know that we have to repeatedly select elements in $\mathfrak{a}$ before finding one that leads to a smooth ideal $\mathfrak{b}$. Hence, we require a randomization process that given the ideal $\mathfrak{a}$, produces as many ideals as required to guarantee to get a smooth $\mathfrak{b}$. This is done by considering ideals of the form $\mathfrak{a} \cdot \prod \mathfrak{p}_i^{e_i}$, where the $\mathfrak{p}_i$ are prime ideals whose norms are below the smoothness bound B.

Then for each ideal $\tilde{\mathfrak{a}} = \mathfrak{a} \cdot \prod \mathfrak{p}_i^{e_i}$, they compute the BKZ_β-reduced basis of the integral ideal $\tilde{\mathfrak{c}} = \mathcal{N}(\tilde{\mathfrak{a}}) \cdot \tilde{\mathfrak{a}}^{-1}$, with a block-size β as determined below. This BKZ-reduction is performed on the ideal lattice $\sigma(\tilde{\mathfrak{c}})$, defined by the canonical embedding of $\tilde{\mathfrak{c}}$. As recalled in Section 1.2.2, it may be viewed as a lattice in $\mathbf{R}^n$ using the Minkowski map.

Denoting by x_v the algebraic integer corresponding to the smallest vector v of the BKZ-reduced basis, they set $\tilde{\mathfrak{b}} = \left\langle \frac{x_v}{\mathcal{N}(\tilde{\mathfrak{a}})} \right\rangle \tilde{\mathfrak{a}}$. Then $\tilde{\mathfrak{b}}$ is in the same class as $\tilde{\mathfrak{a}}$ and

$$\mathcal{N}(\tilde{\mathfrak{b}}) \leq \beta^{\frac{n(n-1)}{2(\beta-1)}} \sqrt{|\Delta_K|}. \tag{4.3}$$

Indeed $\mathcal{N}(\tilde{\mathfrak{c}}) = \mathcal{N}(\tilde{\mathfrak{a}})^{n-1}$ and $\|v\| \leq \beta^{\frac{n-1}{2(\beta-1)}} \mathcal{N}(\tilde{\mathfrak{c}})^{\frac{1}{n}} |\Delta_K|^{\frac{1}{2n}}$ from Theorem 1.1.10 and Lemma 1.4.9. Then $\mathcal{N}(\tilde{\mathfrak{b}}) \leq \left(\frac{\|v\|}{\mathcal{N}(\tilde{\mathfrak{a}})}\right)^n \mathcal{N}(\tilde{\mathfrak{a}})$ leads to the expected result.

If $\tilde{\mathfrak{b}}$ splits over the factor base $\mathcal{B}$, then there exist integers e'_i such that $\tilde{\mathfrak{b}} = \prod \mathfrak{p}_i^{e'_i}$. Thus,

taking care of the randomized factor, we get that the ideal $\frac{x_v}{\mathcal{N}(\tilde{\mathfrak{a}})}\mathfrak{a}$ also splits over $\mathcal{B}$ as $\prod \mathfrak{p}_i^{e_i'-e_i}$. In the end, if $\mathfrak{a}$ splits over $\mathcal{B}$, then the principal ideal

$$\left\langle \frac{x_v}{\mathcal{N}(\mathfrak{a})} \right\rangle \text{ also splits over } \mathcal{B}.$$

Therefore we have derived a relation in the kernel of the surjective morphism defined in Equation (2.5). If $\tilde{\mathfrak{b}}$ does not split, then we try another $\tilde{\mathfrak{a}}$. To bound the number of relations that would be sufficient, they state the following heuristic. Because at least $N = |\mathcal{B}|$ are required, they choose the largest bound that does not increase the complexity.

Heuristic 4.2.2. *There exists a value K that is negligible compared with $|\mathcal{B}|$ such that collecting $K \cdot |\mathcal{B}|$ relations suffices to obtain a relation matrix that has full-rank.*

Finally, using the right parameters and ideals $\mathfrak{a}$ from the factor base, they get the complexities given in Theorem 4.2.1. With a factor base size in $L_{|\Delta_\mathbf{K}|}(a)$ and a block-size $(\log|\Delta_\mathbf{K}|)^a$, the overall complexity turns out to be in $L_{|\Delta_\mathbf{K}|}(a)$.

4.2.2 Our proposition for a simpler algorithm

Instead of precisely studying the complexity of the algorithm of Biasse and Fieker, we rather provide a simpler version, more adapted to our problem. We focus on the relation collection in class group computations for number fields, without using information brought by the defining polynomial. Hence, ideals are viewed as lattices in $\mathbf{R}^n$.

First we do not do the reduction of a specific ideal, but we take as inputs random power-products of factor-base elements. Let $k, A > 0$ be integers in $\mathrm{Poly}(\log|\Delta_\mathbf{K}|)$. We choose k prime ideals $\mathfrak{p}_{j_1}, \ldots, \mathfrak{p}_{j_k}$ in the factor base. For any k-tuple $(e_1, \ldots, e_k) \in \{1, \ldots, A\}^k$, we set $\mathfrak{a} = \prod_{i=1}^k \mathfrak{p}_{j_i}^{e_i}$ and we have

$$\mathcal{N}(\mathfrak{a}) = \mathcal{N}\left(\prod_{i=1}^k \mathfrak{p}_{j_i}^{e_i}\right) \leq \prod_{i=1}^k \mathcal{N}\left(\mathfrak{p}_{j_i}\right)^{e_i} \leq L_{|\Delta_\mathbf{K}|}\left(\beta, c_b\right)^{k\cdot A}.$$

This initialization step can be done by choosing uniformly at random the tuple $(e_1, \ldots, e_k)$ and k prime ideals in $\mathcal{B}$. Since from Landau's Theorem (Theorem 1.5.11), $|\mathcal{B}| = L_{|\Delta_\mathbf{K}|}(\beta, c_b)$, the set of possible samples is large enough for our purposes. In addition, the norm of the input ideals $\mathfrak{a}$ is always polynomial in the size of the factor base.

Second we reduce the lattice defined by the ideal $\mathfrak{a}$ itself, not its normalized inverse. Instead of performing the normalization explained in the previous section, we directly search for a small vector in the ideal $\mathfrak{a}$ — more precisely, in the lattice $\sigma(\mathfrak{a})$ defined by the canonical embedding. Hence we find a small vector v that is the embedding of an algebraic integer x_v.

Because x_v lies in $\mathfrak{a}$, there exists a unique integral ideal $\mathfrak{b}$ such that

$$\langle x_v \rangle = \mathfrak{a}\mathfrak{b}.$$

The attentive reader should point out that the ideals $\mathfrak{a}$ and $\mathfrak{b}$ do not belong to the same ideal class as before. However, this is not so important, because $\mathfrak{b}^{-1}$ for instance shares the same class with $\mathfrak{a}$. Our ultimate goal is to figure out a principal ideal that is B-smooth, and this is achieved with our method too. For the recovery of the algebraic integer, one can make use of the transformation matrix corresponding to the variable change. Another possibility is to work directly with the conjugates and go back to the algebraic representation using round-off, as mentioned in Section 1.2.3.

Third we use the reduction algorithm described by Espitau and Joux in [EJ17]. It works on the Gram matrix of the lattice instead of the basis matrix, and requires less precision. From a practical perspective, their algorithm is able to ensure that the input precision suffices and certifies that the output is an exact reduced basis. A precision analysis similar to the one in Section 3.2 leads to the conclusion that the required precision is polynomial in the size of the input. Indeed, we only have to replace the weight term c^k by the norm of the ideal $L_{|\Delta_\mathbf{K}|}(\beta, c_b)^{k \cdot A}$ whose size is still polynomial in $\log|\Delta_\mathbf{K}|$.

Algorithm 3 Deriving relations from BKZ$_\beta$-reduction

Input: The factor base $\mathcal{B}$, the block-size β, the bounds k and A for building ideals.
Output: The relations stored.

while not enough relations are found **do**
  Choose at random k prime ideals $\mathfrak{p}_{j_1}, \ldots, \mathfrak{p}_{j_k}$ in the factor base $\mathcal{B}$
  Choose at random k exponents $e_{j_1}, \cdots, e_{j_k}$ in $\{1, \ldots, A\}$
  Set $\mathfrak{a} = \prod \mathfrak{p}_i^{e_i}$, for $i \in \{1, \ldots, |\mathcal{B}|\}$, with $e_i = 0$ if $i \notin \{j_1, \ldots, j_k\}$
  Find a BKZ$_\beta$-reduced basis of $\mathfrak{a}$
  Let x_v denote the algebraic integer corresponding to the smallest vector of this basis
  Set $\mathfrak{b}$ as the unique ideal such that $\langle x_v \rangle = \mathfrak{a}\mathfrak{b}$
  if $\mathfrak{b}$ is B-smooth **then**
    Let e'_i such that $\mathfrak{b} = \prod \mathfrak{p}_i^{e'_i}$
    Store the relation $\langle x_v \rangle = \prod \mathfrak{p}_i^{e_i + e'_i}$
  end if
end while

Remark 4.2.3. Another improvement should be to test for smoothness all the elements whose norms are below the bound given by the theoretic study of BKZ reduction. We know that the first vector output by the BKZ reduction has norm below $\beta^{\frac{n-1}{2(\beta-1)}} \mathcal{N}(\mathfrak{a})^{\frac{1}{n}} |\Delta_\mathbf{K}|^{\frac{1}{2n}}$ and this bound is the one we used for the complexity analysis. However, if several small vectors have their norm below this bound, then the rest of the algorithm works similarly for them, and we have saved

the cost of BKZ reductions. Hence, one may try the first small linear combinations between vectors of the reduced basis output after reduction. This is only a practical improvement, because asymptotically the number of BKZ reductions performed is not taken into account (see Section 4.3).

The algorithm stops when enough relations are collected. At this point, it is necessary to rely on a heuristic (as Heuristic 4.2.2) in order to guarantee the result. We propose a new one that suffices for our purposes. We want the number of relations to be sufficient to generate the whole set of relations described in Equation (2.5). We emphasize that there exist ideals in the factor base that are more important: the ones whose norm is below the *Bach bound* $12(\log|\Delta_{\mathbf{K}}|)^2$. Thus we consider that the matrix construction is completed when the number of relations is larger than the number of ideals that occur and when all ideals of norm below the Bach bound are involved in at least one relation. This last condition means that the submatrix built from all the relations and only those ideals must have full-rank. In comparison with Heuristic 4.2.2, our relation matrix may contain all-zero columns, which correspond to ideals in the factor base that are not involved in any of the relations — by construction their norms are necessarily larger than $12(\log|\Delta_{\mathbf{K}}|)^2$.

Heuristic 4.2.4. *There exists K negligible compared with $|\mathcal{B}|$ such that collecting $K\cdot|\mathcal{B}|$ relations suffices to obtain a relation matrix that generates the whole lattice of relations.*

4.2.3 Parameter settings

We consider as input a number field $\mathbf{K} \in \mathcal{D}_{n_0,d_0,\alpha,\gamma}$, with $\alpha \geq \frac{1}{2}$. We stress that no information is needed on the size of the defining polynomial — namely on γ — for this algorithm. Table 2 lists the optimal choices for B and β depending on α, with a transition at $\alpha = \frac{3}{4}$ (as already mentioned by Biasse and Fieker). The parameter $c_b > 0$ is going to be determined later, based on the complexity analysis.

	B	β
$\frac{1}{2} \leq \alpha \leq \frac{3}{4}$	$Ł_{\lvert\Delta_{\mathbf{K}}\rvert}\left(\frac{1}{2}, c_b\right)$	$\left(\log\lvert\Delta_{\mathbf{K}}\rvert\right)^{\frac{1}{2}}$
$\frac{3}{4} < \alpha \leq 1$	$Ł_{\lvert\Delta_{\mathbf{K}}\rvert}\left(\frac{2\alpha}{3}, c_b\right)$	$\left(\log\lvert\Delta_{\mathbf{K}}\rvert\right)^{\frac{2\alpha}{3}}$

Table 2: Optimal choices for the factor base and block-size depending on the extension degree.

4.3 Complexity analyses

4.3.1 The case $\alpha \leq \frac{3}{4}$

According to [BF14], when $\alpha \leq \frac{3}{4}$, we know that our algorithm should run in time $L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}, c_1\right)$. We provide a detailed analysis to find an explicit expression for the constant c_1. Let $\mathbf{K}$ be a number field belonging to $\mathcal{D}_{n_0,d_0,\alpha,\gamma}$ with $\alpha \in \left[\frac{1}{2}, \frac{3}{4}\right]$, $\gamma \geq 1-\alpha$, $d_0 > 0$ and $n_0 > 1$. The factor base $\mathcal{B}$ is fixed as the set of all prime ideals of norm below $B = L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}, c_b\right)$, with $c_b > 0$ to be determined, and the block-size used in BKZ-reduction is $\beta = (\log|\Delta_\mathbf{K}|)^{\frac{1}{2}}$, according to Table 2.

First, we analyze the BKZ-reduction. Before looking at the output, we focus on the cost of the reduction. By construction of ideal $\mathfrak{a}$ — see Section 4.2.2 — its norm is polynomial in $L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}\right)$. Theorem 1.1.10 states that BKZ-reduction runs in time $\mathrm{Poly}\left(n, \log\mathcal{N}(\mathfrak{a})\right) \cdot 2^{O(\beta)}$. Because the norm of $\mathfrak{a}$ is upper bounded, it only remains to bound the factor $2^{O(\beta)}$. Denoting by C the constant in the O, we obtain $\log 2^{O(\beta)} = C \cdot \log 2 \cdot (\log|\Delta_\mathbf{K}|)^{\frac{1}{2}} \leq c(\log|\Delta_\mathbf{K}|)^{\frac{1}{2}}(\log\log|\Delta_\mathbf{K}|)^{\frac{1}{2}}$ asymptotically for any constant $c > 0$. Thus, we have shown that the runtime of the reduction algorithm is below $L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}, c\right)$ for every $c > 0$.

Second, we estimate the norm of the new ideal $\mathfrak{b}$ built from the smallest vector returned by the reduction algorithm. From Theorem 1.1.10 and Lemma 1.4.9, we know that the smallest vector v of the BKZ_β-reduced basis has a norm that satisfies $\|v\| \leq \beta^{\frac{n-1}{2(\beta-1)}}\mathcal{N}(\mathfrak{a})^{\frac{1}{n}}|\Delta_\mathbf{K}|^{\frac{1}{2n}}$. As $\mathcal{N}(x_v) \leq \|v\|^n$ (Lemma 1.4.2), we directly derive that the norm of $\mathfrak{b}$ is upper bounded by $\mathcal{N}(\mathfrak{b}) \leq \beta^{\frac{n(n-1)}{2(\beta-1)}}\sqrt{|\Delta_\mathbf{K}|}$ so that we deduce[1]

$$\begin{aligned}
\log\mathcal{N}(\mathfrak{b}) &\leq \frac{1}{2}\log|\Delta_\mathbf{K}| + \frac{n_0^2}{4}(\log|\Delta_\mathbf{K}|)^{2\alpha-\frac{1}{2}}(\log\log|\Delta_\mathbf{K}|)^{1-2\alpha} \\
&\leq \frac{1}{2}\log|\Delta_\mathbf{K}| + c(\log|\Delta_\mathbf{K}|)^{2\alpha-\frac{1}{2}}(\log\log|\Delta_\mathbf{K}|)^{1-2\alpha+\frac{1}{2}} \qquad \text{for all } c > 0 \\
&\leq \frac{1}{2}\log|\Delta_\mathbf{K}|\,(1+o(1)) \\
\implies \quad \mathcal{N}(\mathfrak{b}) &\leq L_{|\Delta_\mathbf{K}|}\left(1, \frac{1}{2}\right).
\end{aligned}$$

Third, we have to express the probability for $\mathfrak{b}$ to be B-smooth. Assuming Heuristic 1.4.20 allows us to get a probability of

$$L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}, \frac{1}{4c_b}\right)^{-1}.$$

Hence, on average, testing $L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}, \frac{1}{4c_b}\right)$ ideals $\mathfrak{a}$ leads to a single ideal $\mathfrak{b}$ that is B-smooth and thus to one relation. Assuming Heuristic 4.2.4, we need to find $L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}, c_b\right)$ relations. This

[1] Note that it is the same bound as in Equation (4.3). Our adjustments in the algorithm do not affect this bound.

requires testing

$$L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}, \frac{1}{4c_b} + c_b\right)$$

ideals for smoothness. Proposition 1.4.18 states that each test costs $L_{L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)}\left(\frac{1}{2}\right) = L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{4}\right)$, which is negligible. The reduction step, whose runtime is below $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}, c\right)$ for every $c > 0$, is also negligible. Hence the global complexity of the relation collection step is given by the number of ideals that we test, that is

$$L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}, \frac{1}{4c_b} + c_b\right).$$

Complexity for the class group computation. Now that we know the complexity of the collection step, we look at the remaining parts of the computation to get the class group structure, in order to determine the best c_b. The relations are stored in a matrix of size $K \cdot N \times N$, with $N = |\mathcal{B}| = L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}, c_b\right)$. The results regarding linear algebra, precision and regulator computation are already studied by Biasse and Fieker and recalled in Section 2.3.3. They show [BF14, Proposition 4.1] that the class group structure is inferred from the relation matrix in time $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}, (\omega+1)c_b\right)$, where ω denotes the matrix multiplication exponent. This result essentially relies on the HNF algorithm of Storjohann, recalled in Section 1.1.1.

The best choice for c_b — *i.e.*, the one that minimizes the complexity — follows from balancing the runtimes of the collection and linear algebra phases. Thus the parameter $c_b > 0$ should satisfy

$$\frac{1}{4c_b} + c_b = (\omega+1)c_b \quad \Longleftrightarrow \quad c_b = \frac{1}{2\sqrt{\omega}}.$$

Theorem 4.3.1. *Assuming ERH and Heuristics 1.4.20 and 4.2.4, for every number field* $\mathbf{K}$ *that belongs to* $\mathcal{D}_{n_0,d_0,\alpha,\gamma}$ *with* $\alpha \in \left[\frac{1}{2}, \frac{3}{4}\right]$*, our algorithm computes the class group structure and the regulator with runtime*

$$L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}, \frac{\omega+1}{2\sqrt{\omega}}\right).$$

Remark 4.3.2. We recall that ω denotes the exponent arising in the complexity of matrix multiplication. The smallest known value is $\omega = 2.3728639$ (see [Gal14]) which correspond to the value 1.095 for the second constant. In practice, we use the Strassen algorithm [Str69] where $\omega = \log_2 7 \approx 2.807$, leading to the second-constant value 1.136.

4.3.2 The case $\alpha > \frac{3}{4}$

We follow the same path as in the previous case although some adjustments are made. We start by mentioning that our final complexity is much better than the one announced in [BF14]: we manage to replace the first constant $\frac{2}{3} + \varepsilon$ by $\frac{2\alpha}{3}$, which is always smaller, particularly when α

is close to $\frac{3}{4}$. Furthermore, our second constant can be chosen arbitrarily small, which we denote by $L_{|\Delta_\mathbf{K}|}\left(\frac{2\alpha}{3}, o(1)\right)$.

This time, $\mathbf{K}$ belongs to $\mathcal{D}_{n_0,d_0,\alpha,\gamma}$ with $\alpha \in \left(\frac{3}{4}, 1\right]$. The smoothness bound is fixed to $B = Ł_{|\Delta_\mathbf{K}|}\left(\frac{2\alpha}{3}, c_b\right)$, $c_b > 0$, and the block-size is $\beta = (\log|\Delta_\mathbf{K}|)^{\frac{2\alpha}{3}}$. The bound on the norms $\mathcal{N}(\mathfrak{a})$ is polynomial in $L_{|\Delta_\mathbf{K}|}\left(\frac{2\alpha}{3}\right)$, because of the parameters we used for constructing the ideals $\mathfrak{a}$. In the same way as in Section 4.3.1, we show that the runtime of the reduction algorithm is below $L_{|\Delta_\mathbf{K}|}\left(\frac{2\alpha}{3}, c\right)$ for every $c > 0$. The bound we derive for the norm of the new ideal built is

$$\begin{aligned}
\log\mathcal{N}(\mathfrak{b}) &\leq \frac{1}{2}\log|\Delta_\mathbf{K}| + \frac{\alpha n_0^2}{3}(\log|\Delta_\mathbf{K}|)^{\frac{4\alpha}{3}}(\log\log|\Delta_\mathbf{K}|)^{1-2\alpha} \\
&\leq \frac{1}{2}\log|\Delta_\mathbf{K}| + c(\log|\Delta_\mathbf{K}|)^{\frac{4\alpha}{3}}(\log\log|\Delta_\mathbf{K}|)^{1-\frac{4\alpha}{3}} \qquad \text{for all } c > 0 \\
&\leq c(\log|\Delta_\mathbf{K}|)^{\frac{4\alpha}{3}}(\log\log|\Delta_\mathbf{K}|)^{1-\frac{4\alpha}{3}} \qquad \text{for all } c > 0.
\end{aligned}$$

Assuming Heuristic 1.4.20 and fixing any $c > 0$, if we take $c_b = \sqrt{\frac{2\alpha c}{3}}$ in the definition of B, then the probability for ideal $\mathfrak{b}$ to be B-smooth is

$$L_{|\Delta_\mathbf{K}|}\left(\frac{2\alpha}{3}, c_b\right)^{-1}.$$

Hence we conclude that testing $L_{|\Delta_\mathbf{K}|}\left(\frac{2\alpha}{3}, 2c_b\right)$ ideals suffices for the entire collection phase. Again, the runtime $L_{|\Delta_\mathbf{K}|}\left(\frac{\alpha}{3}\right)$ to perform a single smoothness test can be neglected.

Complexity for the class group computation. The global complexity of the class group computation follows directly, because the runtime of the linear algebra step is obtained by multiplying the second constant $2c_b$ by a constant factor $\omega + 1$. As the constant c_b could be chosen arbitrarily small (but positive), we get the following theorem.

Theorem 4.3.3. *Assuming ERH and Heuristics 1.4.20 and 4.2.4, for every number field* $\mathbf{K}$ *that belongs to* $\mathcal{D}_{n_0,d_0,\alpha,\gamma}$ *with* $\alpha \in \left(\frac{3}{4}, 1\right]$*, our algorithm computes the class group structure and the regulator with runtime*

$$L_{|\Delta_\mathbf{K}|}\left(\frac{2\alpha}{3}, o(1)\right).$$

We can now update Figure 1, by taking into account the results of Theorems 4.3.1 and 4.3.3. This is presented in Figure 2.

4.4 Using HNF to get an even smaller complexity

We have a complexity between $L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}\right)$ and $L_{|\Delta_\mathbf{K}|}\left(\frac{2}{3}\right)$ that grows linearly for classes $\mathcal{D}$ with $\alpha \geq \frac{3}{4}$. We want to reduce this worst case using Cheon's trick, recalled in Section 1.1.3. As

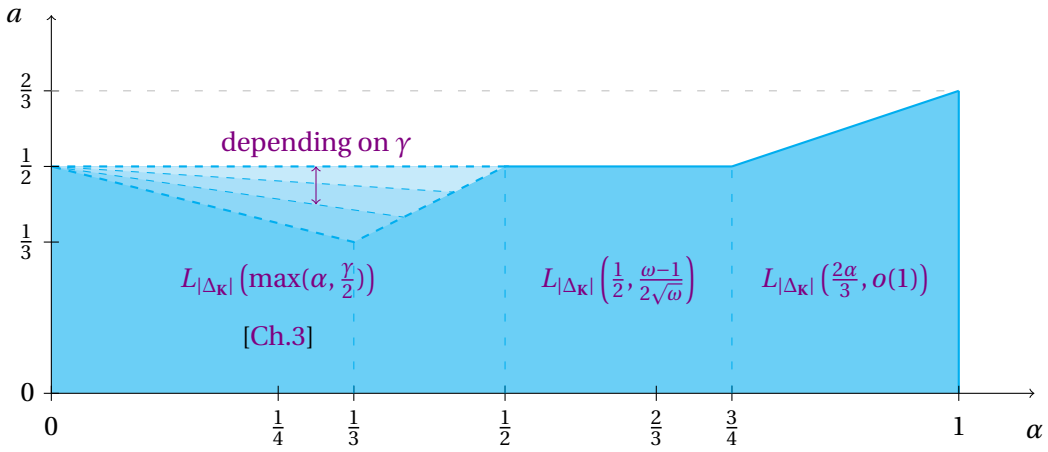

Figure 2: Complexity obtained by our algorithms.

we have seen, it allows to output a shorter vector than in the general case. It relies on the reduction of a sublattice that has smaller dimension than the full lattice, provided that the input lattice has small discriminant.

As shown in this section, the complexity we are able to reach with this method is $L_{|\Delta_K|}\left(\frac{2\alpha+1}{5}\right)$. It varies linearly between $L_{|\Delta_K|}\left(\frac{1}{2}\right)$ and $L_{|\Delta_K|}\left(\frac{3}{5}\right) < L_{|\Delta_K|}\left(\frac{2}{3}\right)$. Hence, we fix the smoothness bound $B = Ł_{|\Delta_K|}\left(\frac{2\alpha+1}{5}, c_b\right)$, with $c_b > 0$ to be determined. Also, the block-size used for BKZ-reductions is set to $\beta = \left(\log|\Delta_K|\right)^{\frac{2\alpha+1}{5}}$. Overall, the path followed by this improved version of our algorithm is essentially similar to the one described in Section 4.2.2. We only mention the adjustments in the rest of the section.

First, we need to work with an integral lattice. Indeed as we begin by computing the HNF of the lattice, it must be defined over $\mathbf{Z}$. This is not a problem, as we already mentioned. We know that the required precision is polynomial in the size of the entries. Practically, we approximate the Gram matrix and use the implementation of [EJ17].

Second, to ensure that the hypothesis of Corollary 1.1.12 is satisfied, we need a bound on the determinant of the input lattice. As we want a lattice with *small determinant* as input, we first perform a rough reduction, using the classical BKZ algorithm — that is without Cheon's trick. Given an ideal $\mathfrak{a}$ constructed as above as a power-product of elements in the factor base and denoting by v the first vector of the BKZ-reduced basis, we define the ideal $\mathfrak{b}$ as the unique integral ideal that satisfies

$$\langle x_v \rangle = \mathfrak{a}\mathfrak{b}.$$

Thanks to the analysis presented in Section 4.3.2, we know that the norm of this ideal $\mathfrak{b}$ is upper bounded by $L_{|\Delta_K|}\left(\frac{8\alpha-1}{5}\right)$. We are in the case $\alpha > \frac{3}{4}$, so that $\frac{8\alpha-1}{5} > 1$. The determinant of the lattice corresponding to the canonical embedding of $\mathfrak{b}$ is $\mathcal{N}(\mathfrak{b}) \cdot \sqrt{|\Delta_K|}$. Hence we cannot

expect that this quantity is smaller than $L_{|\Delta_\mathbf{K}|}(1)$, so we look for an ideal $\mathfrak{b}$ that is $\widetilde{B}$-smooth for $\widetilde{B} = Ł_{|\Delta_\mathbf{K}|}(1,1)$. According to Proposition 1.4.18, each smoothness test costs

$$L_{L_{|\Delta_\mathbf{K}|}(1)}\left(\frac{1}{2}\right) = L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}\right)$$

and assuming Heuristic 1.4.20, testing $L_{|\Delta_\mathbf{K}|}\left(\frac{8\alpha-6}{5}\right)$ ideals suffices on average. In addition, the number of ideals in every smooth decomposition is bounded by $\left(\log|\Delta_\mathbf{K}|\right)^{\frac{8\alpha-6}{5}}\left(1+o(1)\right)$. The complete runtime of this smoothness phase is in $L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}\right)$, as $0 < \frac{8\alpha-6}{5} < \frac{2}{5}$, which is outweighed by the initial BKZ$_\beta$ reduction, whose cost is $L_{|\Delta_\mathbf{K}|}\left(\frac{2\alpha+1}{5}, o(1)\right)$.

In the end, we have ideals $\mathfrak{b}_1,\ldots,\mathfrak{b}_l$ whose canonical embeddings have determinant in $Ł_{|\Delta_\mathbf{K}|}(1,1) < \beta^{\frac{n^2}{2\beta}} = L_{|\Delta_\mathbf{K}|}\left(\frac{8\alpha-1}{5}\right)$ and which satisfy $\prod\mathfrak{b}_i = \mathfrak{b}$. We notice that for the application of Corollary 1.1.12, the lower bound $Ł_{|\Delta_\mathbf{K}|}(1,1)$ does not have to be reached. However, as the quality of the output relies on this quantity — a factor $\log_\beta \det\mathcal{L}$ appears in the exponent — we minimize it in order to get the best possible output.

For each ideal lattice $\sigma(\mathfrak{b}_i)$, we may apply Cheon's trick combined with BKZ-reduction. As for all i it is the case that $\log(\det\sigma(\mathfrak{b}_i)) = C\log|\Delta_\mathbf{K}|$ for a constant $C>0$, a small vector v_i is found with norm satisfying

$$\begin{aligned}
\log\|v_i\| &\leq \left(\frac{2C\log|\Delta_\mathbf{K}|}{\left(\log|\Delta_\mathbf{K}|\right)^{\frac{2\alpha+1}{5}}\log\left(\left(\log|\Delta_\mathbf{K}|\right)^{\frac{2\alpha+1}{5}}\right)}\right)^{\frac{1}{2}}\log\left(\left(\log|\Delta_\mathbf{K}|\right)^{\frac{2\alpha+1}{5}}\right)\left(1+o(1)\right)\\
&\leq \sqrt{\frac{2C(2\alpha+1)}{5}}\left(\log|\Delta_\mathbf{K}|\right)^{\frac{2-\alpha}{5}}\left(\log\log|\Delta_\mathbf{K}|\right)^{\frac{1}{2}}\left(1+o(1)\right)\\
&\leq c\left(\log|\Delta_\mathbf{K}|\right)^{\frac{2-\alpha}{5}}\left(\log\log|\Delta_\mathbf{K}|\right)^{1-\frac{2-\alpha}{5}} \qquad \text{for every constant } c>0.
\end{aligned}$$

Again, as we already did for the earlier analyses, we bound the norm of the algebraic integer x_{v_i} associated to the vector v_i using Lemma 1.4.2. We obtain the inequality $\mathcal{N}\left(\langle x_{v_i}\rangle\right) \leq L_{|\Delta_\mathbf{K}|}\left(\frac{4\alpha+2}{5}, c\right)$ for every $c>0$. In addition, there exist integral ideals $\mathfrak{c}_i$ such that $\langle x_{v_i}\rangle = \mathfrak{b}_i\mathfrak{c}_i$ for all i. As the norm of $\mathfrak{b}_i$ is less than $Ł_{|\Delta_\mathbf{K}|}(1,1)$, we deduce that for each i, the norm of the ideal $\mathfrak{c}_i$ satisfies

$$\mathcal{N}(\mathfrak{c}_i) \leq L_{|\Delta_\mathbf{K}|}\left(\frac{4\alpha+2}{5}, o(1)\right).$$

Denoting by c the arbitrarily small non-negative constant that arises in the $o(1)$, we follow the same argument as in Section 4.3.2. By fixing $c_b = \sqrt{\frac{(2\alpha+1)c}{5}}$, we deduce that the probability for each $\mathfrak{c}_i$ to be B-smooth is

$$L_{|\Delta_\mathbf{K}|}\left(\frac{2\alpha+1}{5}, c_b\right)^{-1}.$$

Hence we conclude that testing $L_{|\Delta_\mathbf{K}|}\left(\frac{2\alpha+1}{5}, 2c_b\right)$ ideals suffices to complete the relation

collection. Indeed, we have to test $L_{|\Delta_\mathbf{K}|}\left(\frac{2\alpha+1}{5}, c_b\right)$ ideals for each ideal $\mathfrak{b}_i$ and given an ideal $\mathfrak{b}$ as input, the number of factors $\mathfrak{b}_i$ is polynomial. Finally, assuming Heuristic 4.2.4, we require $L_{|\Delta_\mathbf{K}|}\left(\frac{2\alpha+1}{5}, c_b\right)$ relations, which leads to the runtime stated above for the relation collection.

Complexity for the class group computation. Again, as in Section 4.3.2, the final complexity for the class group computation follows directly and we get the following theorem.

Theorem 4.4.1. *Assuming ERH and Heuristics 1.4.20 and 4.2.4, for every number field* $\mathbf{K}$ *that belongs to* $\mathcal{D}_{n_0,d_0,\alpha,\gamma}$ *with* $\alpha \in \left(\frac{3}{4}, 1\right]$*, our algorithm computes the class group structure and the regulator with runtime*

$$L_{|\Delta_\mathbf{K}|}\left(\frac{2\alpha+1}{5}, o(1)\right).$$

This new result allows to reduce the slope of the increasing line appearing in our complexity figures. The worst complexity now becomes $L_{|\Delta_\mathbf{K}|}\left(\frac{3}{5}, o(1)\right)$. This result is displayed in Figure 3.

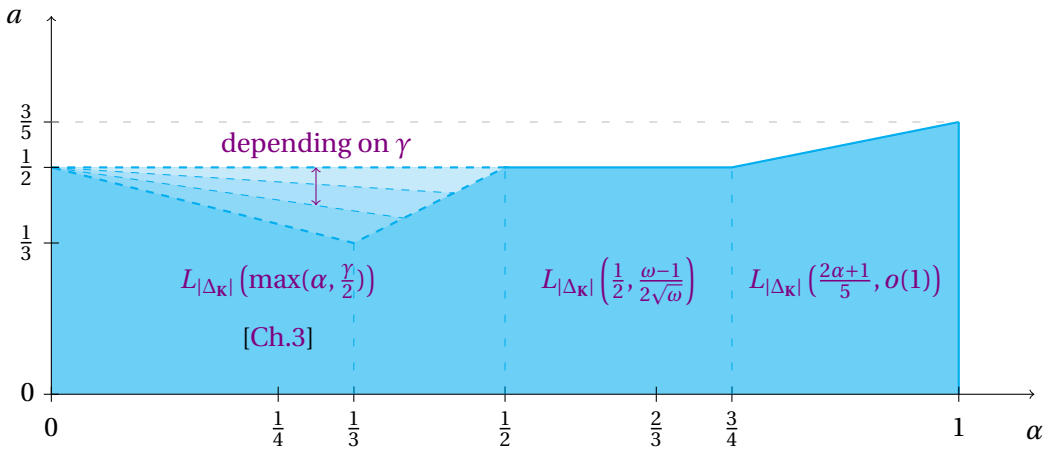

Figure 3: Complexity obtained by our algorithms and Cheon's trick ($\alpha \leq \frac{1}{2} \Rightarrow 1-\alpha \leq \gamma \leq 1$).

Chapter 5

Reducing the complexity using good defining polynomials

Contents

In this chapter, we focus on a conditional improvement based on the smallness of the defining polynomial. Though ideal-reduction schemes enforce an $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)$ complexity, the solution of the discrete logarithm problem in finite fields in $L_Q\left(\frac{1}{3}\right)$ suggests that we can reach this value for class group computations too. This is the aim of the *sieving strategy*. We first describe the algorithm and extend the results obtained by Biasse in [Bia14a]. Then we study its complexity, compare it with the results of Chapter 4 and exhibit the number fields for which this new strategy offers a smaller complexity than ideal reductions. In addition, we provide an algorithm for solving the Principal Ideal Problem by using our class group computations.

5.1 Motivation

We have already seen that computing class groups and regulators in number fields is essentially based on the index calculus method. Within this strategy, the part that determines the complexity is the relation collection, because the linear-algebra step only leads to an additional constant factor in the exponent — *i.e.*, in the second constant in the L-notation. The relation collection step, as its name suggests, consists in searching for many principal ideals that split over the factor base:

$$\langle x \rangle \mathcal{O}_{\mathbf{K}} = \prod \mathfrak{p}_i^{e_i}.$$

As described in Chapter 4, in the general case, without making any assumption on the number fields, the ideal-reduction strategy performs best and leads to a complexity that is at least $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)$. However, there exist conditional improvements when the number field is defined by a *good* polynomial, that is a polynomial having small height. Indeed, in that case, the $\mathfrak{q}$-descent strategy described by Biasse and Fieker in [BF14] and generalized in Chapter 3 allows a complexity between $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{3}\right)$ and $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)$ for all number fields belonging to a class $\mathcal{D}$ with $\alpha \leq \frac{1}{2}$.

Our new idea that underlies this chapter is to generate the relations by testing a lot of *small* principal ideals that are generated by algebraic integers of bounded degree and coefficients. The norms of such elements depend on the two bounds used for the degree and on the coefficients and the height of the defining polynomial — which is the reason why we want the smallest possible defining polynomial. This idea was already used in the Number Field Sieve, as recalled in Section 2.4.2. Enge, Gaudry, and Thomé [EG07, EGT11] extend this method to low-degree curves for solving the discrete logarithm problem in the groups of such curves in $L_{q^g}\left(\frac{1}{3}\right)$, where q is the cardinality of the base field and g the genus of the curve.

Then, Biasse in [Bia14a] applies the method in the context of class group computations.

His result only addresses very specific number fields $\mathbf{K}$ defined by a polynomial T such that

$$\left[\mathbf{K}:\mathbf{Q}\right] \leq O(\log|\Delta_{\mathbf{K}}|)^{\alpha} \quad \text{and} \quad \log H(T) \leq O(\log|\Delta_{\mathbf{K}}|)^{1-\alpha} \tag{5.1}$$

for an α in the open interval $\left(\frac{1}{3},\frac{2}{3}\right)$. In so doing, he was able to compute the class group in time $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{3}\right)$ assuming ERH and under heuristics. We generalize here the sieving strategy to all number fields, obtaining a complexity possibly as low as $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{3}\right)$.

This method has also been used by Buchmann, Jacobson, Neis, Theobald, and Weber in [BJN+99] for practical enhancements. Indeed, the sieving strategy definitely outperforms ideal reduction in practice, especially for small-degree number fields.

The $\mathfrak{q}$-descent strategy explained in [BF14], where elements with small coefficients are searched in lattices of smaller dimension, is, in a certain sense, another way to use these small algebraic integers. However, our method appears easier to understand and its complexity analysis is streamlined: we are able to provide explicitly the second constant in the L-notation, which does not sound that simple for the $\mathfrak{q}$-descent. In addition, from a practical point of view, as the $\mathfrak{q}$-descent only works in small degree $\left(\alpha \leq \frac{1}{2}\right)$, our algorithm should outperform the $\mathfrak{q}$-descent, since it does not require iterations nor lattice-reductions.

5.2 Deriving relations by sieving

In the following, we make use of the classification presented in Chapter 4. For a fixed number field $\mathbf{K}$ in a class $\mathcal{D}_{n_0,d_0,\alpha,\gamma}$, the value $\alpha \in [0,1]$ corresponds to the extension degree so that it is precisely defined. For the second main parameter $\gamma \geq 1-\alpha$, special care should be taken: sometimes it costs too much to reduce the defining polynomial. This issue is addressed in Section 5.4: given a number field defined by a polynomial, we study the optimal strategy for computing the class group depending on the parameters. Is the polynomial reduction necessary ? Is it better to use ideal-reduction or sieving ?

Remark 5.2.1. We use the terminology *"sieving strategy"* because it closely corresponds to the way to — efficiently — implement it. Theoretically, our algorithm only consists in testing for smoothness a huge arithmetic progression of algebraic integers until we have found sufficiently many relations.

The description of the algorithm we are going to introduce is clear and the algorithm is easily understandable. Difficulties arise when we need to fix the parameters such as the smoothness bound for the factor base and the bounds that describe the sieving space in order to minimize the complexity. To fix the notation, we consider a number field $\mathbf{K} = \mathbf{Q}(\theta)$ of degree n and let T denote the defining polynomial of which θ is a root.

Let $B > 0$ be the smoothness bound that must be determined. We fix the factor base $\mathcal{B} = \{\mathfrak{p}_1, \ldots, \mathfrak{p}_N\}$ as the set of all prime ideals of $\mathcal{O}_\mathbf{K}$ whose norm is below B. From the Landau Prime Ideal Theorem (Theorem 1.5.11), we know that its cardinality satisfies

$$N = |\mathcal{B}| = B\,(1 + o(1))\,.$$

We describe the sieving space by fixing a bound $t > 0$ on the degree, together with a bound $S > 0$ on the coefficients. Hence we use all the polynomials of degree at most t with coefficients between $-S$ and S. These are $(2S+1)^{t+1}$ polynomials, but only half of them are of interest, as algebraic integers x and $-x$ generate the same ideal. Note that we may also avoid algebraic integers built from a reducible polynomial in θ. Indeed, if $x = x_1 \cdot x_2$, then the exponents of a relation produced by x equal the sums of the exponents of relations produced by x_1 and x_2.

Given an algebraic integer $x = \sum_{i=0}^{t} a_i \theta^i$ and denoting by A the polynomial $A(X) = \sum_{i=0}^{t} a_i X^i$, Equations (1.12) and (1.11) imply that the norm of the principal ideal $\langle x \rangle$ is given by

$$\mathcal{N}\big(\langle x \rangle\big) = \mathrm{Res}\,(A, T)\,.$$

From the two bounds t and S, we derive an upper bound for the norm of the principal ideal $\langle x \rangle$, using Lemma 1.4.4:

$$\mathcal{N}\big(\langle x \rangle\big) \leq \sqrt{t+1}^{\,n} \sqrt{n+1}^{\,t} H(T)^t S^n. \tag{5.2}$$

Assuming Heuristic 1.4.20, this bound on the norm offers a lower bound on the probability $\mathcal{P}$ of B-smoothness of any principal ideal $\langle x \rangle$ belonging to the sieving space. Then the $(2S+1)^{t+1}$ small ideals lead to $(2S+1)^{t+1} \cdot \mathcal{P}$ relations. Assuming Heuristic 4.2.4, collecting $N\big(1+o(1)\big)$ relations suffices to derive the class group. Therefore we want the following relation to be satisfied by our choice of parameters:

$$(2S+1)^{t+1} \cdot \mathcal{P} = N\big(1 + o(1)\big). \tag{5.3}$$

Remark 5.2.2. Note that making use of our weaker Heuristic 4.2.4, introduced in Chapter 4, is essential here. Indeed, the factor base may contain ideals of degree $k > t$, that cannot be part of any relations derived from our settings. Because every ideal whose norm is below the Bach bound has a degree smaller than $\log 12 + 2 \log\log |\Delta_\mathbf{K}|$, we know that sieving on degree-t polynomials suffices for our purposes, which was not the case with the prior heuristic.

To evaluate the cost of the sieving phase, we need to know the number of ideals we test for smoothness: it is $(2S+1)^{t+1}$. We explain below that the cost of each smoothness test is always

negligible. Then the overall cost of the sieving phase is given by $(2S+1)^{t+1}(1+o(1))$.

We have already stated that the lowest final complexity is obtained when a balance is reached between the cost of the relation collection and the cost of the linear-algebra phase. Thus we also want that

$$(2S+1)^{t+1} = N^{\omega+1}\big(1+o(1)\big), \tag{5.4}$$

because the linear algebra cost is in $N^{\omega+1}$ (see Section 4.3).

Before determining the parameters that minimize the complexity, we give an outline of the strategy in Algorithm 4.

Algorithm 4 Deriving relations from small algebraic integers

Input: The factor base $\mathcal{B}$, the degree bound t and the coefficient bound S.
Output: The relations stored.

for d **from** 1 **to** t **do**
  for all $(a_0,\ldots,a_d) \in [-S,\ldots,S]^{d+1}$ **do**
    Fix $x = \sum a_i\theta^i$ and $\mathfrak{a} = \langle x\rangle$
    Test the B-smoothness of $\mathfrak{a}$
    if $\mathfrak{a}$ is B-smooth **then**
      Fix e_i such that $\mathfrak{a} = \prod \mathfrak{p}_i^{e_i}$
      Store the relation $\langle x\rangle = \prod \mathfrak{p}_i^{e_i}$
    end if
  end for
end for

We describe in the subsequent sections how to set the parameters for the factor base and the sieving space to achieve the best complexities. We fix $n_0 > 1$, $d_0 > 0$, $\alpha \in [0,1]$ and $\gamma \geq 1-\alpha$ and let $\mathbf{K}$ be a number field that belongs to $\mathcal{D}_{n_0,d_0,\alpha,\gamma}$. We also assume that we know a *good* defining polynomial T that satisfies

$$\log H(T) \quad \leq \quad d_0(\log|\Delta_{\mathbf{K}}|)^{\gamma}(\log\log|\Delta_{\mathbf{K}}|)^{1-\gamma}.$$

Let θ be a primitive element of $\mathbf{K}$ that is a root of the defining polynomial T. As in the discrete logarithm problem in finite fields, we need to distinguish several cases according to the relative sizes of α and γ. However, the distinctions between the various cases are not as precise as they are for the DLP: we consider small, medium and large degrees and give the corresponding inequalities involving α and γ.

5.3 Complexity analyses

5.3.1 The case of medium degree

We begin by the medium case, which we define by α and γ being of the same magnitude. This includes $\alpha \approx \gamma \approx \frac{1}{2}$, but covers a much wider range as follows from the analysis below. As already discussed at the beginning of Section 4.1, the size of the defining polynomial plays a role in the complexity: we only have the inequality $\gamma \geq 1-\alpha$, so that we have no choice but to keep using both α and γ.

Given that we hope to find an algorithm with runtime $L_{|\Delta_\mathbf{K}|}\left(\frac{1}{3}\right)$ and given that $\gamma \geq 1-\alpha$ (thus $\alpha+\gamma \geq 1$), we simply conjecture the existence of an algorithm with runtime $L_{|\Delta_\mathbf{K}|}\left(\frac{\alpha+\gamma}{3}\right)$ and fix the size of the factor base $\mathcal{B}$ as the set of prime ideals of norm at most

$$B = Ł_{|\Delta_\mathbf{K}|}\left(\frac{\alpha+\gamma}{3}, c_b\right),$$

with $c_b > 0$ to be determined. Thanks to Theorem 1.5.11, we know that $N = |\mathcal{B}| = L_{|\Delta_\mathbf{K}|}\left(\frac{\alpha+\gamma}{3}, c_b\right)$. The sieving space is chosen to consist in all algebraic integers $x = A(\theta)$, built as polynomials in θ, that satisfy

$$\deg A \leq t = c_t\left(\frac{\log|\Delta_\mathbf{K}|}{\log\log|\Delta_\mathbf{K}|}\right)^{\frac{2}{3}(\alpha+\gamma)-\gamma} \quad \text{and} \quad H(A) \leq S = Ł_{|\Delta_\mathbf{K}|}\left(\frac{2}{3}(\alpha+\gamma)-\alpha, c_s\right). \tag{5.5}$$

In particular, $\log H(A) \leq c_s(\log|\Delta_\mathbf{K}|)^{\frac{2}{3}(\alpha+\gamma)-\alpha}(\log\log|\Delta_\mathbf{K}|)^{1-\left(\frac{2}{3}(\alpha+\gamma)-\alpha\right)}$. Of course, these two quantities are only well defined for $\frac{2}{3}(\alpha+\gamma)-\gamma \geq 0$ and $\frac{2}{3}(\alpha+\gamma)-\alpha \geq 0$, which defines the bounds of the medium-degree case.

According to Equation (5.2), this choice of parameters enables to bound the norm of every principal ideal $\langle x\rangle$ in the sieving space by

$$\mathcal{N}\big(\langle x\rangle\big) \leq L_{|\Delta_\mathbf{K}|}\left(\frac{2}{3}(\alpha+\gamma), n_0c_s + d_0c_t\right). \tag{5.6}$$

We deduce from Heuristic 1.4.20 that a principal ideal generated by such an x is B-smooth with probability

$$\mathcal{P} \geq L_{|\Delta_\mathbf{K}|}\left(\frac{\alpha+\gamma}{3}, \frac{(\alpha+\gamma)(n_0c_s+d_0c_t)}{3c_b}\right)^{-1}.$$

The size of the sieving space is given by $(2S+1)^{t+1} = Ł_{|\Delta_\mathbf{K}|}\left(\frac{\alpha+\gamma}{3}, c_sc_t\right)$. As usual, this estimation allows us to estimate the number of relations found by combining the two previous

results: the number of collected relations is expected to be

$$(2S+1)^{t+1} \cdot \mathcal{P} = L_{|\Delta_{\mathbf{K}}|}\left(\frac{\alpha+\gamma}{3}, c_s c_t - \frac{(\alpha+\gamma)(n_0 c_s + d_0 c_t)}{3c_b}\right).$$

With $N = L_{|\Delta_{\mathbf{K}}|}\left(\frac{\alpha+\gamma}{3}, c_b\right)$ and the assumption of Heuristic 4.2.4 (see Equation (5.3)), we obtain

$$c_s c_t - \frac{(\alpha+\gamma)(n_0 c_s + d_0 c_t)}{3c_b} = c_b.$$

Another equation between the various constants stems from the balance between the relation collection and the linear algebra, as stated by Equation (5.4). It boils down to

$$c_s c_t = (\omega+1)c_b.$$

From these two equations, we easily express c_t in the other constants and obtain a degree-2 equation in c_b, depending on c_s: $3\omega c_s c_b^2 - d_0(\alpha+\gamma)(\omega+1)c_b - n_0(\alpha+\gamma)c_s^2 = 0$. This expression allows us to infer the shape of c_b, which is going to give us the final complexity, depending on c_s:

$$c_b = \frac{d_0(\alpha+\gamma)(\omega+1) + \sqrt{d_0^2(\alpha+\gamma)^2(\omega+1)^2 + 12n_0(\alpha+\gamma)\omega c_s^3}}{6\omega c_s}.$$

It only remains to minimize this quantity as a function of c_s. It follows from a straight analysis that the minimum is achieved for c_s satisfying $c_s^3 = \frac{2d_0^2(\alpha+\gamma)(\omega+1)^2}{3n_0\omega}$, which leads to

$$c_b = \left(\frac{4n_0 d_0(\alpha+\gamma)^2(\omega+1)}{9\omega^2}\right)^{\frac{1}{3}}.$$

Consequently, the runtime of our algorithm for computing the class group structure and an approximation of the regulator is

$$L_{|\Delta_{\mathbf{K}}|}\left(\frac{\alpha+\gamma}{3}, \left(\frac{4n_0 d_0(\alpha+\gamma)^2(\omega+1)^4}{9\omega^2}\right)^{\frac{1}{3}}\right).$$

Remark 5.3.1. The first constant may be as low as $\frac{1}{3}$ if γ reaches the lower bound $1-\alpha$, *i.e.*, $\alpha+\gamma = 1$.

We also mention that in this case, our second constant is better than the one found by Biasse in [Bia14a].

This analysis however only holds when the two quantities $\frac{2}{3}(\alpha+\gamma)-\gamma$ and $\frac{2}{3}(\alpha+\gamma)-\alpha$ are non-negative. These conditions offer the limits of our analysis and can be rewritten as

$$\frac{1}{3}(\alpha+\gamma) \leq \alpha \leq \frac{2}{3}(\alpha+\gamma) \qquad \Longleftrightarrow \qquad \frac{\gamma}{2} \leq \alpha \leq 2\gamma.$$

Therefore, it remains to treat the two complementary cases, when either the size of the defining-polynomial height or the extension degree prevails.

5.3.2 The small-degree case: when $2\alpha < \gamma$

The first extreme case we study is when the size of the defining-polynomial height outweighs the extension degree. It corresponds to the left part of the diagrams displayed in Chapter 4, where the $\mathbf{q}$-descent strategy works. In these cases, the extension degree satisfies

$$\alpha < \frac{\gamma}{2} \qquad \Longleftrightarrow \qquad \alpha < \frac{1}{3}(\alpha + \gamma).$$

We are able to reach a final complexity in $L_{|\Delta_\mathbf{K}|}\left(\frac{\gamma}{2}\right)$ for the relation collection. As α is relatively small — below $\frac{\gamma}{2}$ — we know that the defining-polynomial reduction algorithm presented in Chapter 3 runs in time $L_{|\Delta_\mathbf{K}|}(\alpha)$, which is strictly less than $L_{|\Delta_\mathbf{K}|}\left(\frac{\gamma}{2}\right)$. Hence this reduction is always negligible compared with the relation collection, so that it can be considered as a precomputation. According to Corollary 3.1.3 and the discussion at the end of Section 3.1, we can also assume $\gamma \leq 1$.

We fix the size of the factor base $\mathcal{B}$ by considering all the prime ideals having norm below

$$B = Ł_{|\Delta_\mathbf{K}|}\left(\frac{\gamma}{2}, c_b\right),$$

and we have from Landau's theorem that $N = |\mathcal{B}| = L_{|\Delta_\mathbf{K}|}\left(\frac{\gamma}{2}, c_b\right)$. The sieving space is constructed as before, using all polynomials A that satisfy

$$\deg A \leq t = c_t \quad \text{and} \quad H(A) \leq S = Ł_{|\Delta_\mathbf{K}|}\left(\frac{\gamma}{2}, c_s\right). \tag{5.7}$$

These adjustments in the definition are motivated by the desire to minimize the norm size. As the height of the defining polynomial is large, we bound the degree of the algebraic integers to guarantee that the norm stays small.

According to Equation (5.2), this choice of parameters enables to bound the norm of every principal ideal $\langle x \rangle$ in the sieving space by

$$\mathcal{N}\big(\langle x \rangle\big) \leq L_{|\Delta_\mathbf{K}|}\left(\gamma, d_0 c_t\right). \tag{5.8}$$

Assuming Heuristic 1.4.20 allows us to have the following inequality satisfied by the probability for a principal ideal generated by such an x to be B-smooth:

$$\mathcal{P} \geq L_{|\Delta_\mathbf{K}|}\left(\frac{\gamma}{2}, \frac{d_0 \gamma c_t}{2c_b}\right)^{-1}.$$

As the sieving-space cardinality is $(2S+1)^{t+1} = Ł_{|\Delta_\mathbf{K}|}\left(\frac{\gamma}{2}, c_s(c_t+1)\right)$, we obtain the number of collected relations as before and Equation (5.3) results in $c_s(c_t+1) - \frac{d_0\gamma c_t}{2c_b} = c_b$. Similarly Equation (5.4) leads to $c_s(c_t+1) = (\omega+1)c_b$. From an identical approach as in the previous section, we find the optimal choices for the constants and conclude that the runtime of our algorithm is

$$L_{|\Delta_\mathbf{K}|}\left(\frac{\gamma}{2}, \left(\frac{d_0\gamma(\omega+1)^2 c_t}{2\omega}\right)^{\frac{1}{2}}\right).$$

Remark 5.3.2. The first constant is always between $\frac{1}{3}$ and $\frac{1}{2}$: the upper bound is a consequence of the precomputation made for finding the minimal-height defining polynomial while the lower one comes from $\gamma > \frac{2}{3}(\alpha+\gamma) \geq \frac{2}{3}$. In the second constant, the factor c_t appears so that the complexity depends on the degree of the polynomials we use for sieving. The minimal value is obtained for $c_t = 1$, for a runtime in $L_{|\Delta_\mathbf{K}|}\left(\frac{\gamma}{2}, \left(\frac{d_0\gamma(\omega+1)^2}{2\omega}\right)^{\frac{1}{2}}\right)$.

Remark 5.3.3. A possible alternative for the sieving may be to enlarge the sieving space by allowing the coefficients to be larger — always below $S' = Ł_{|\Delta_\mathbf{K}|}\left(\gamma-\alpha, o(1)\right)$ — and to consider only a random subset of size $L_{|\Delta_\mathbf{K}|}\left(\frac{\gamma}{2}, c_s(c_t+1)\right)$ of the sieving space. Using the bound S' does not affect Equation (5.8) and the complexity is preserved.

5.3.3 The large-degree case: when $\alpha > 2\gamma$

In this last case, the extension degree outweighs the size of the defining-polynomial height. It corresponds to the right part of the diagrams displayed in Chapter 4. Here we have to work with the input defining polynomial because finding the minimal one costs too much. As the extension degree is large, we opt for sieving polynomials that have small coefficients and large degrees.

We fix the size of the factor base $\mathcal{B}$ by considering all the prime ideals having norm below

$$B = Ł_{|\Delta_\mathbf{K}|}\left(\frac{\alpha}{2}, c_b\right),$$

and we have from Landau's theorem that $N = |\mathcal{B}| = L_{|\Delta_\mathbf{K}|}\left(\frac{\alpha}{2}, c_b\right)$. The sieving space is constructed using all polynomials A that satisfy

$$\deg A \leq t = c_t\left(\frac{\log|\Delta_\mathbf{K}|}{\log\log|\Delta_\mathbf{K}|}\right)^{\frac{\alpha}{2}} \quad \text{and} \quad H(A) \leq S = Ł_{|\Delta_\mathbf{K}|}(0, c_s) = (\log|\Delta_\mathbf{K}|)^{c_s}. \tag{5.9}$$

According to Equation (5.2), this choice of parameters enables to bound the norm of every principal ideal $\langle x\rangle$ in the sieving space by

$$\mathcal{N}\left(\langle x\rangle\right) \leq L_{|\Delta_\mathbf{K}|}\left(\alpha, n_0\left(c_s + \frac{\alpha}{4}\right)\right). \tag{5.10}$$

We deduce from Equation (5.10) and Heuristic 1.4.20 that the probability for a principal ideal generated by such an x to be B-smooth satisfies

$$\mathcal{P} \geq L_{|\Delta_{\mathbf{K}}|}\left(\frac{\alpha}{2}, \frac{n_0\alpha(\alpha+4c_s)}{8c_b}\right)^{-1}.$$

Finally, an identical analysis enables to find the optimal choice for the constants. The final runtime for our class group algorithm based on sieving strategy satisfies

$$L_{|\Delta_{\mathbf{K}}|}\left(\frac{\alpha}{2}, \left(\frac{n_0\alpha(\alpha+4c_s)(\omega+1)^2}{8\omega}\right)^{\frac{1}{2}}\right).$$

Remark 5.3.4. The first constant is always between $\frac{1}{3}$ and $\frac{1}{2}$ because $\alpha > \frac{2}{3}(\alpha+\gamma) \geq \frac{2}{3}$. In the second constant, the constant c_s appears which can be chosen arbitrarily small. The minimal runtime thus becomes $L_{|\Delta_{\mathbf{K}}|}\left(\frac{\alpha}{2}, \left(\frac{n_0\alpha^2(\omega+1)^2}{8\omega}\right)^{\frac{1}{2}}\right)$.

Remark 5.3.5. Again, it is possible to enlarge the sieving space by allowing the degree to be larger — always below $t' = c_t\left(\frac{\log|\Delta_{\mathbf{K}}|}{\log\log|\Delta_{\mathbf{K}}|}\right)^{\alpha-\gamma-\varepsilon}$ for $\varepsilon > 0$ arbitrarily small — and to consider only a random subset of the sieving space of size $L_{|\Delta_{\mathbf{K}}|}\left(\frac{\alpha}{2}, c_s c_t\right)$. Using the bound t' does not affect Equation (5.10) and the complexity is preserved.

5.4 Conclusion on sieving strategy

The complexity analyses we have derived in the previous sections assume that we know a small defining polynomial T, that is a witness to the fact that $\mathbf{K}$ belongs to the class $\mathcal{D}$. We recall that the classes $\mathcal{D}$ satisfy

$$\mathcal{D}_{n_0,d_F,\alpha,\gamma_F} \quad \subset \quad \mathcal{D}_{n_0,d_0,\alpha,\gamma_0},$$

for $n_0, d_0, d_F > 0$, $0 \leq \alpha \leq 1$ and $1-\alpha \leq \gamma_F < \gamma_0$. To identify the best strategy depending on the inputs, we consider a number field $\mathbf{K}$ defined by a polynomial T such that

$$\frac{1}{n_0}\left(\frac{\log|\Delta_{\mathbf{K}}|}{\log\log|\Delta_{\mathbf{K}}|}\right)^{\alpha} \leq \deg T \leq n_0\left(\frac{\log|\Delta_{\mathbf{K}}|}{\log\log|\Delta_{\mathbf{K}}|}\right)^{\alpha} \text{ and } \log H(T) \leq d_0(\log|\Delta_{\mathbf{K}}|)^{\gamma_0}(\log\log|\Delta_{\mathbf{K}}|)^{1-\gamma_0}.$$

It is easily verified that $\mathbf{K}$ belongs to $\mathcal{D}_{n_0,d_0,\alpha,\gamma_0}$. In addition we introduce γ_F and d_F so that γ_F is the minimal γ such that $\mathbf{K} \in \mathcal{D}_{n_0,d_F,\alpha,\gamma}$. Thus we consider two different classes to which $\mathbf{K}$ belongs, namely $\mathcal{D}_{n_0,d_F,\alpha,\gamma_F}$ and $\mathcal{D}_{n_0,d_0,\alpha,\gamma_0}$; note that

$$\mathbf{K} \in \mathcal{D}_{n_0,d_F,\alpha,\gamma_F} \subset \mathcal{D}_{n_0,d_0,\alpha,\gamma_0}.$$

Given the number field $\mathbf{K}$ defined by the polynomial T as inputs, we study the different options for computing the class group and give the optimal strategy. Let us first look at the medium-degree case, where $\frac{\gamma_0}{2} \leq \alpha \leq 2\gamma_0$. Necessarily, we have $\alpha \geq \frac{\alpha+\gamma_0}{3} \geq \frac{1}{3}$.

- When $\alpha \leq \frac{1}{2}$, as $\gamma_0 \leq 2\alpha$, we have $\frac{\alpha+\gamma_0}{3} \leq \frac{1}{2}$ and sieving is the best strategy.
- When $\frac{1}{2} < \alpha \leq \frac{3}{4}$, the sieving strategy remains optimal as long as $\frac{\alpha+\gamma_0}{3} \leq \frac{1}{2}$. Indeed, beyond this bound, the ideal-reduction strategy becomes less costly and should be preferred. This happens as soon as $\gamma_0 \geq 1$.
- Similarly, for $\frac{3}{4} < \alpha \leq 1$, the sieving strategy remains optimal as long as $\frac{\alpha+\gamma_0}{3} \leq \frac{2\alpha+1}{5}$. Above this bound, the ideal-reduction strategy becomes the best option. This happens as soon as $\gamma_0 \geq \frac{4}{5}$.

The large-degree case is easier to deal with. Provided that $\alpha > 2\gamma_0$, we know that the sieving strategy results in an algorithm with runtime $L_{|\Delta_\mathbf{K}|}\left(\frac{\alpha}{2}\right)$, between $L_{|\Delta_\mathbf{K}|}\left(\frac{1}{3}\right)$ and $L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}\right)$, as $\alpha > 2\gamma_0$ implies that $\alpha \geq \frac{2(\alpha+\gamma_0)}{3} \geq \frac{2}{3}$. This is always the best option.

The small-degree case is when defining-polynomial reduction plays a role. Indeed, we know that its cost is $L_{|\Delta_\mathbf{K}|}(\alpha)$ while the sieving strategy runs in time $L_{|\Delta_\mathbf{K}|}\left(\frac{\gamma}{2}\right)$. Because $\alpha < \frac{\gamma_0}{2}$, we can always perform this reduction as a precomputation. It allows to find the smallest-height defining polynomial and so the minimal γ_F. This reduction has two outcomes:

- If $\frac{\gamma_F}{2} < \alpha$, then the sieving strategy has a complexity in $L_{|\Delta_\mathbf{K}|}\left(\frac{\alpha+\gamma_F}{3}\right)$, which is negligible compared to the cost of the reduction, so that the final runtime is $L_{|\Delta_\mathbf{K}|}(\alpha)$. This can only happens when $\alpha > \frac{1}{3}$, since $\alpha + \gamma_F \geq 1$.
- If $\frac{\gamma_F}{2} > \alpha$, then the sieving strategy has a complexity that outweighs the cost of the reduction, so that the final runtime is $L_{|\Delta_\mathbf{K}|}\left(\frac{\gamma_F}{2}\right)$. This value is between $L_{|\Delta_\mathbf{K}|}\left(\frac{1}{3}\right)$ and $L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}\right)$, as the reduction algorithm returns a polynomial such that $\gamma_F \leq 1$ — this is a direct consequence of Corollary 3.1.3, already stated in Chapter 3 — and because $\gamma_F > 2\alpha$ implies that $\gamma_F \geq \frac{2(\alpha+\gamma_0)}{3} \geq \frac{2}{3}$. This is the only option when $\alpha < \frac{1}{3}$.

The results of this analysis are summarized in Table 3. We also give a new diagram for the complexities in Figure 4.

5.5 Application to Principal Ideal Problem

In addition to the step forward for class group computations, our results allow us to improve the resolution of another problem: the Principal Ideal Problem (PIP). It consists in finding a generator of an ideal, assuming it is principal. The Short Principal Ideal Problem (SPIP) follows

Cond. on α	**Cond. on γ**	**Strategy**	**Complexity**
$\alpha \leq \frac{1}{2}$	$\gamma_0 \leq 2\alpha$	Sieving (MD)	$L_{\lvert\Delta_K\rvert}\left(\frac{\alpha+\gamma_0}{3}\right)$
	$2\alpha < \gamma_F \leq \gamma_0$	Pol. Red. & Sieving (SD)	$L_{\lvert\Delta_K\rvert}\left(\frac{\gamma_F}{2}\right)$
	$\gamma_F < 2\alpha < \gamma_0$	Pol. Red. & Sieving (SD)	$L_{\lvert\Delta_K\rvert}(\alpha)$
$\alpha > \frac{1}{2}$	$2\gamma_0 \leq \alpha$	Sieving (LD)	$L_{\lvert\Delta_K\rvert}\left(\frac{\alpha}{2}\right)$
	$\frac{\alpha+\gamma_0}{3} \leq \max\left(\frac{1}{2}, \frac{2\alpha+1}{5}\right)$	Sieving (MD)	$L_{\lvert\Delta_K\rvert}\left(\frac{\alpha+\gamma_0}{3}\right)$
	$\frac{\alpha+\gamma_0}{3} > \max\left(\frac{1}{2}, \frac{2\alpha+1}{5}\right)$	Ideal Reduction	$L_{\lvert\Delta_K\rvert}\left(\max\left(\frac{1}{2}, \frac{2\alpha+1}{5}\right)\right)$

Table 3: Choice of the strategy depending on the input parameters.

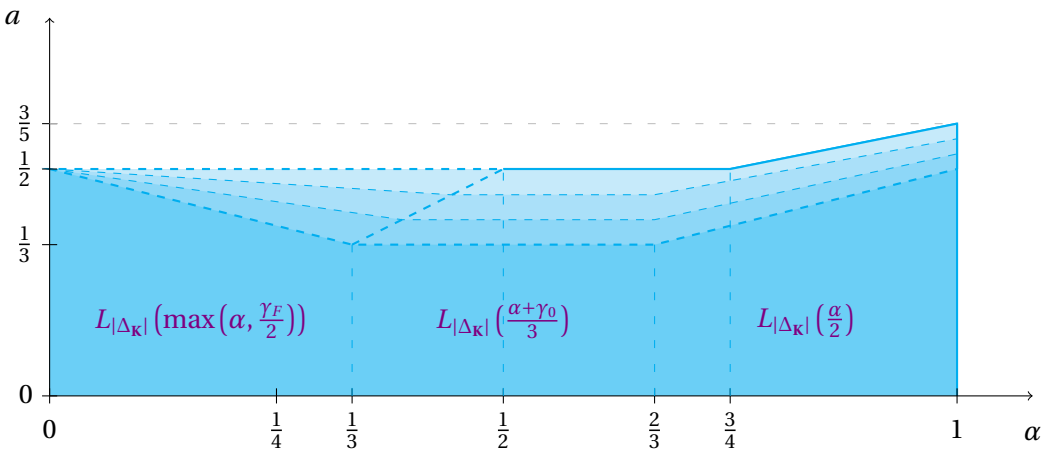

Figure 4: Complexity obtained with our sieving strategy.

from the PIP by adding the assumption that there exists a *small* generator. The SPIP is the base of several Fully Homomorphic Encryption schemes inspired by the work of Gentry [Gen09] such as the FHE scheme presented by Smart and Vercauteren at PKC 2010 [SV10] and the multilinear map scheme presented by Garg, Gentry, and Halevi at EuroCrypt in 2013 [GGH13]. Solving the SPIP is a two-stage process that consists of first solving the underlying PIP (on which we focus here), if successful followed by attempts to reduce the generator found to a short one (this issue is postponed to Chapter 6). Finding a generator of a principal ideal, and even testing the principality of an ideal, are difficult problems in algorithmic number theory, as described in detail in [Coh93, Chapter 4] and [Thi95, Section 7].

The general strategy is — again — similar to the one used for the Discrete Logarithm Problem in finite fields. Indeed, for finding the logarithm of an element, two steps are distinguished: first, we find the logarithms of many small elements; second, we express our target element using these small elements and recover its logarithm. It is the same here with our

ideal $\mathfrak{a}$, assumed to be principal. First, we compute the matrix of relations as for class group computations, keeping track of the small elements we have sieved with. Second, we find an ideal $\mathfrak{b}$ that is in the same class as $\mathfrak{a}$ and that splits over the factor base. Then, linear algebra allows us to recover a generator of $\mathfrak{b}$ thanks to the relation matrix and finally, we can solve the PIP.

5.5.1 The descent algorithm

We first briefly outline the algorithm without fixing the parameters. Indeed, as for class group computations, the optimal parameters choices are derived from the complexity analyses, depending on the number-field exponents α and γ. In order to bootstrap the descent, we start with a classical BKZ-reduction to obtain an ideal of reasonable norm. Indeed, as the input ideal $\mathfrak{a}$ is fixed — the one for which we want a generator — it can have an arbitrarily large norm. All the ideal reductions are performed on the lattice built from the coefficient embedding $\varsigma(\mathfrak{a})$. The block-size is fixed so that the complexity of the reduction is strictly below the overall complexity of the algorithm, as we have done in Chapter 4. Then the descent consists in a succession of ideal reductions and smoothness tests so that the norms of all ideals involved decrease progressively until they reach the lower bound, given by the smoothness bound used in the class group computations.

We now fix the parameters for a degree-n number field $\mathbf{K}$ that belongs to a class $\mathcal{D}_{n_0,d_0,\alpha,\gamma}$ with $\frac{\gamma}{2} \leq \alpha \leq 2\gamma$. We know that the final complexity is given by $L_{|\Delta_\mathbf{K}|}\left(\frac{\alpha+\gamma}{3}\right)$, assuming this first constant is small enough — below $\frac{1}{2}$. Let us write $k = \frac{\alpha+\gamma}{3}$ for the sake of simplicity. A pattern of the descent is displayed in Figure 5.

The initial reduction. Let $\mathfrak{a}$ be the ideal, assumed principal, for which we search for a generator. We may also assume that it is prime, otherwise it suffices to factor it and to work with the prime ideals — that have smaller norms. We can always represent this ideal with its HNF. We obtain an $n \times n$ matrix whose largest coefficient is at most the norm of the ideal $\mathcal{N}(\mathfrak{a})$.

The first reduction consists in performing a BKZ-reduction on the n-dimensional lattice $\varsigma(\mathfrak{a})$ with block-size $\beta = (\log|\Delta_\mathbf{K}|)^k$. It permits to exhibit a small vector v that satisfies $\|v\| \leq \beta^{\frac{n-1}{2(\beta-1)}}\mathcal{N}(\mathfrak{a})^{\frac{1}{n}}$, as $\det\varsigma(\mathfrak{a}) = \mathcal{N}(\mathfrak{a})$ (see Theorem 1.1.10). The cost of this lattice reduction is $L_{|\Delta_\mathbf{K}|}\big(k, o(1)\big)$, provided that the norm $\mathcal{N}(\mathfrak{a})$ satisfies $\log\mathcal{N}(\mathfrak{a}) \leq L_{|\Delta_\mathbf{K}|}(k-\varepsilon)$ for $\varepsilon > 0$, as specified in Section 4.3. Therefore, the principal ideal generated by the algebraic integer $x_0 \in \mathfrak{a}$ corresponding to the vector $v \in \varsigma(\mathfrak{a})$ has its norm bounded by $(n+1)^n \cdot H(T)^n \cdot \beta^{\frac{n(n-1)}{2(\beta-1)}}\mathcal{N}(\mathfrak{a})$ (see Corollary 1.4.7). Finally, denoting by $\mathfrak{a}^{(0)}$ the unique integral ideal such that $\langle x_0\rangle = \mathfrak{a}\cdot\mathfrak{a}^{(0)}$,

we obtain the following upper bound:

$$\mathcal{N}\left(\mathfrak{a}^{(0)}\right) \leq L_{|\Delta_{\mathbf{K}}|}\left(\alpha+\gamma, n_0 d_0\right) = L_{|\Delta_{\mathbf{K}}|}\left(3k, n_0 d_0\right).$$

As we have mentioned, we alternate lattice reductions and smoothness tests. For keeping a complexity in $L_{|\Delta_{\mathbf{K}}|}(k)$, we are going to test the ideal $\mathfrak{a}^{(0)}$ for $L_{|\Delta_{\mathbf{K}}|}(2k, s_0)$-smoothness, for $s_0 > 0$ to be determined. According to Proposition 1.4.18, the cost for a single test is $L_{|\Delta_{\mathbf{K}}|}\left(k, \sqrt{2ks_0}\right)$, while the assumption of Heuristic 1.4.20 asserts that the probability for $\mathfrak{a}^{(0)}$ to be $L_{|\Delta_{\mathbf{K}}|}(2k, s_0)$-smooth is lower bounded by $L_{|\Delta_{\mathbf{K}}|}\left(k, \frac{kn_0d_0}{s_0}\right)^{-1}$. First, this implies that we need to test on average $L_{|\Delta_{\mathbf{K}}|}(k)$ ideals before finding one that is smooth. We then make use of the randomization process used by Biasse and Fieker in [BF14] and recalled in Section 4.2.1. It consists in considering randomized ideals that are products of $\mathfrak{a}$ with random power-products of small prime ideals — the ones in the factor base. Clearly, it offers sufficiently many choices for testing $L_{|\Delta_{\mathbf{K}}|}(k)$ ideals. Second, the total runtime for the smoothness tests is given by

$$L_{|\Delta_{\mathbf{K}}|}\left(k, \frac{kn_0d_0}{s_0} + \sqrt{2ks_0}\right),$$

which is minimal for $s_0^3 = 2k(n_0d_0)^2$, leading to a complexity of

$$L_{|\Delta_{\mathbf{K}}|}\left(k, \left(\frac{9}{2}k^2 n_0 d_0\right)^{\frac{1}{3}}\right).$$

Subsequent steps. At the beginning of the i-th step, we have an ideal $\mathfrak{a}^{(i)}$ whose norm is upper bounded by $L_{|\Delta_{\mathbf{K}}|}\left(k\left(1+\frac{1}{2^i}\right), s_i\right)$. This time, we are going to perform the lattice reduction over a sublattice of $\varsigma\left(\mathfrak{a}^{(i)}\right)$ of dimension $d = c_d\left(\frac{\log|\Delta_{\mathbf{K}}|}{\log\log|\Delta_{\mathbf{K}}|}\right)^{\delta}$, for $0 \leq \delta \leq \alpha$ and $c_d > 0$ to be determined. The reason to look at a sublattice is that it allows to reduce the norms of the ideals that are involved, which is exactly what we want for the descent.

The BKZ-reduction on this sublattice provides an algebraic integer $x_i \in \mathfrak{a}^{(i)}$ and so an integral ideal $\mathfrak{a}^{(i+1)}$ such that $\langle x_i\rangle = \mathfrak{a}^{(i)} \cdot \mathfrak{a}^{(i+1)}$. The upper bound we get on the norm of $\mathfrak{a}^{(i+1)}$, according to Theorem 1.1.10 and Lemma 1.4.4, is

$$L_{|\Delta_{\mathbf{K}}|}(\alpha) \cdot L_{|\Delta_{\mathbf{K}}|}\left(\gamma+\delta, d_0c_d\right) \cdot L_{|\Delta_{\mathbf{K}}|}(\alpha+\delta-k) \cdot L_{|\Delta_{\mathbf{K}}|}\left(\alpha + k\left(1+\frac{1}{2^i}\right) - \delta, \frac{n_0 s_i}{c_d}\right).$$

This quantity is minimal when $\gamma+\delta = \alpha + k\left(1+\frac{1}{2^i}\right) - \delta \iff \delta = \alpha - k\left(1+\frac{1}{2^{i+1}}\right)$ and $c_d^2 = \frac{n_0 s_i}{d_0}$, which results in the following upper bound for the norm:

$$L_{|\Delta_{\mathbf{K}}|}\left(k\left(2+\frac{1}{2^{i+1}}\right), 2\sqrt{n_0 d_0 s_i}\right).$$

Again, we want to test this ideal for smoothness and we fix the smoothness bound to $L_{|\Delta_\mathbf{K}|}\left(k\left(1+\frac{1}{2^{i+1}}\right), s_{i+1}\right)$. This time, the cost for a single ECM is negligible, as given by $L_{|\Delta_\mathbf{K}|}\left(\frac{k}{2}\left(1+\frac{1}{2^{i+1}}\right)\right)$. The total cost is then inferred from the number of ideals we have to test. Using the same process as for the initial reduction and assuming Heuristic 1.4.20, this number is

$$L_{|\Delta_\mathbf{K}|}\left(k, \frac{2k\sqrt{n_0 d_0 s_i}}{s_{i+1}}\right).$$

The final step. We fix $l = \left\lceil \log_2\left(\frac{1}{k}\log\left(\frac{\log|\Delta_\mathbf{K}|}{\log\log|\Delta_\mathbf{K}|}\right)\right)\right\rceil$. Thus, at step l, we have ideals that are $L_{|\Delta_\mathbf{K}|}\left(k\left(1+\frac{1}{2^l}\right), s_l\right)$-smooth. However, by definition of the L-notation,

$$\log L_{|\Delta_\mathbf{K}|}\left(k\left(1+\frac{1}{2^l}\right), s_l\right) \leq s_l(\log|\Delta_\mathbf{K}|)^k(\log\log|\Delta_\mathbf{K}|)^{1-k}\underbrace{\left(\frac{\log|\Delta_\mathbf{K}|}{\log\log|\Delta_\mathbf{K}|}\right)^{1/\log\left(\frac{\log|\Delta_\mathbf{K}|}{\log\log|\Delta_\mathbf{K}|}\right)}}_{= \; e=\exp(1)}\big(1+o(1)\big),$$

so that we have the inequality $L_{|\Delta_\mathbf{K}|}\left(k\left(1+\frac{1}{2^l}\right), s_l\right) \leq L_{|\Delta_\mathbf{K}|}(k, e\cdot s_l)$.

Remark 5.5.1. More precisely, we can go further and get rid of the constant e. Indeed, for every $\varepsilon > 0$, if C_ε denotes the smallest integer larger than $\log(1+\varepsilon)^{-1}$, then at step $C_\varepsilon \cdot l$, we only consider ideals that are $L_{|\Delta_\mathbf{K}|}(k, (1+\varepsilon)s_l)$-smooth.

In the end, we want all the ideals involved to have a norm below the smoothness bound we have used for class group computation, *i.e.*,

$$e\cdot s_l \leq c_b = \left(\frac{4k^2 n_0 d_0(\omega+1)}{\omega^2}\right)^{\frac{1}{3}}. \tag{5.11}$$

Our approach is to balance the cost of all steps, except the initial one: each one costs $L_{|\Delta_\mathbf{K}|}\left(k, (4k^2 n_0 d_0 y)^{\frac{1}{3}}\right)$, for a constant $y > 0$ to be determined. Hence we have, for all i,

$$\frac{2k\sqrt{n_0 d_0 s_i}}{s_{i+1}} = 4k^2 n_0 d_0 y \iff s_{i+1} = \sqrt{s_i}\cdot\left(\frac{4k^2 n_0 d_0}{y^2}\right)^{\frac{1}{6}}.$$

We deduce that

$$\begin{aligned} s_l &= s_0^{\frac{1}{2^l}}\cdot\left(\frac{4k^2 n_0 d_0}{y^2}\right)^{\frac{1}{6}\cdot\left(1+\frac{1}{2}+\cdots+\frac{1}{2^{l-1}}\right)} \\ &= \left(\frac{s_0 y^{\frac{2}{3}}}{(4k^2 n_0 d_0)^{\frac{1}{3}}}\right)^{\frac{1}{2^l}}\left(\frac{4k^2 n_0 d_0}{y^2}\right)^{\frac{1}{3}} \\ &= \left(\frac{4k^2 n_0 d_0}{y^2}\right)^{\frac{1}{3}}\big(1+o(1)\big). \end{aligned}$$

Then, Equation (5.11) can be rewritten as $e\left(\frac{4k^2 n_0 d_0}{y^2}\right)^{\frac{1}{3}} \leq \left(\frac{4k^2 n_0 d_0(\omega+1)}{\omega^2}\right)^{\frac{1}{3}}$, *i.e.*, $y^2 \geq \frac{e^3\omega^2}{\omega+1}$. As the number of steps is polynomial in $\log|\Delta_{\mathbf{K}}|$, the total cost of the l steps of the descent is $L_{|\Delta_{\mathbf{K}}|}\left(k, \left(4k^2 n_0 d_0 y\right)^{\frac{1}{3}}\right)$, with $y^2 = \frac{e^3\omega^2}{\omega+1}$. It outweighs the initial reduction, because $4y > \frac{9}{2}$ for $\omega \geq 2$.

Remark 5.5.2. We need to bound the numbers of ideals involved in order to be sure of our final complexity. At each step, we spend time $L_{|\Delta_{\mathbf{K}}|}(k)$ for the smoothness tests. It follows that the number of ideals in the decomposition is bounded by $O\left(\left(\frac{\log|\Delta_{\mathbf{K}}|}{\log\log|\Delta_{\mathbf{K}}|}\right)^k\right)$. During the descent, the number of ideals is then multiplied by this factor at each step. Finally, the number of ideals at step l is quasi-polynomial $O\left(\left(\frac{\log|\Delta_{\mathbf{K}}|}{\log\log|\Delta_{\mathbf{K}}|}\right)^k\right)^l$. In Figure 5, indices have been added to the ideals to illustrate this.

At this point, the only remaining part consists in finding out how to decompose these ideals over the principal ideals collected for building the relation matrix. This is done by solving a linear system $MX = Y$, where M is the relation matrix and Y the valuations vector of the smooth ideal. To be sure that this system has a solution, we need to have a relation matrix of *almost-full* rank. By this unusual term, we only mean that we want all ideals in the factor base involved in the relations, except the ones whose degree is larger than the bound c_t. Indeed, they do not appear in a relation because of the parameters we use, but we do not care as they do not arise either in the descent process — this is a consequence of the dimensions of the sublattices that we use. The runtime of this part is $L_{|\Delta_{\mathbf{K}}|}(k, 2c_b)$ as the matrix of relations is already in HNF.

Finally, we also have $y < \frac{(\omega+1)^4}{\omega^2}$, which means that the complexity for solving the Principal Ideal Problem is the same as the complexity obtained for class group computation. However, we have analyzed the runtime of the descent for the case when the matrix of relations is known.

Remark 5.5.3. Two improvements can be made to reduce the complexity. First, as explained in Remark 5.5.1, the constant e can be replaced by any other constant larger than and arbitrarily close to 1. Second, if we are only interested in solving the PIP, then the computation of the regulator and the class group structure are useless. Hence, the linear-algebra step boils down to solving a linear system over $\mathbf{Z}$, which can be performed in time $L_{|\Delta_{\mathbf{K}}|}(k, \omega c_b)$ using a Las-Vegas algorithm described by Storjohann in [Sto05]. Then, we can adjust all our parameters replacing $\omega + 1$ by ω. Finally, these enhancements lead to a final complexity for the PIP of

$$L_{|\Delta_{\mathbf{K}}|}\left(k, \left(\frac{4k^2 n_0 d_0 \omega^4}{(\omega-1)^2}\right)^{\frac{1}{3}}\right).$$

Remark 5.5.4. The descent strategy for solving the Principal Ideal Problem is also treated in detail in Chapter 6. It is applied in the context of the cryptanalysis of a Fully Homomorphic

Encryption Scheme over prime-power cyclotomic fields. The interested reader can find more details there.

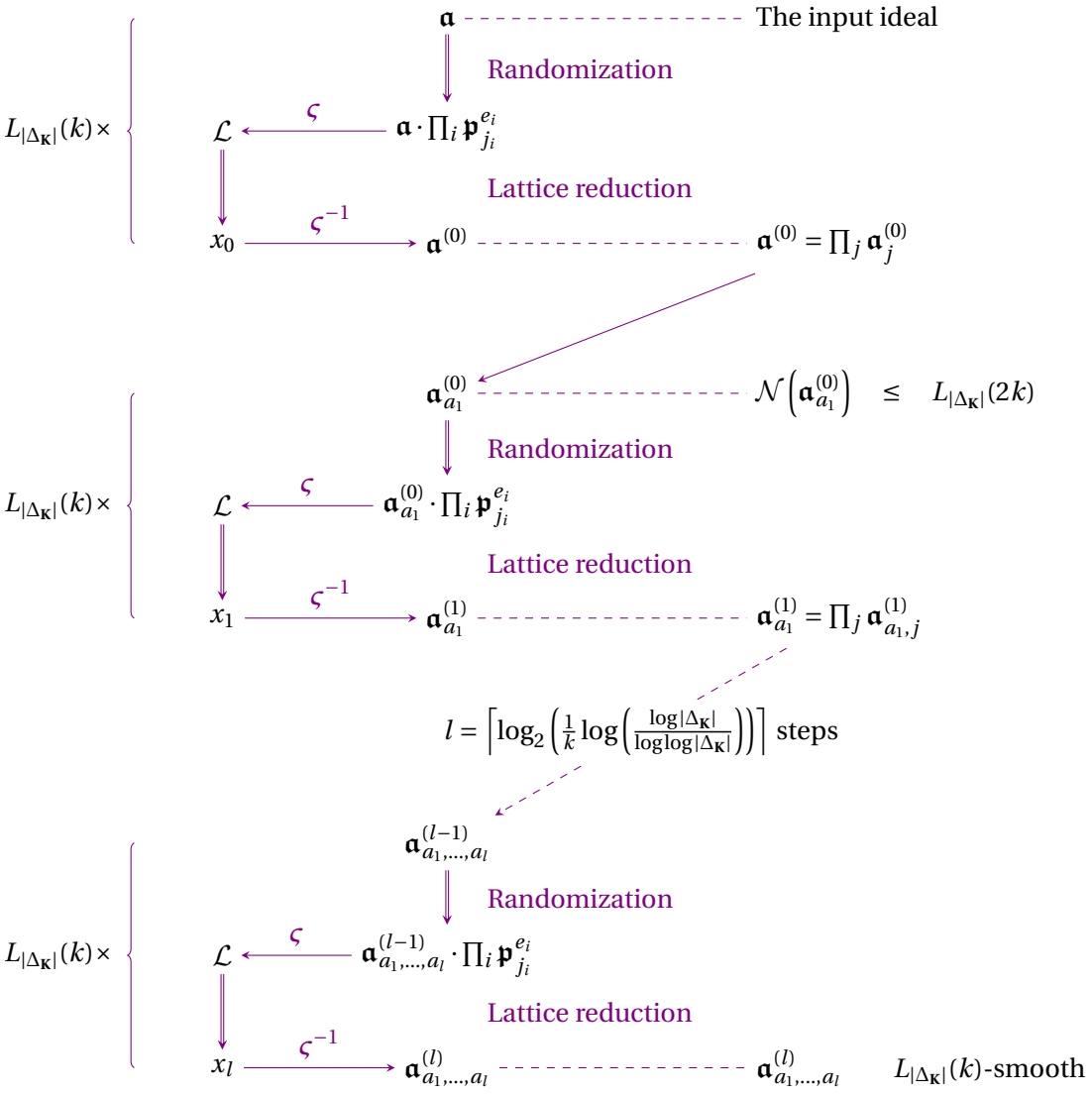

Figure 5: The descent algorithm for the medium-degree case.

5.5.2 The large-degree case

For the present large-degree case, the approach is similar to the previous case, the only difference being the parameters choice. This time, $\alpha > 2\gamma$ and we denote by k the first constant of the class group complexity, *i.e.*, $k = \frac{\alpha}{2}$.

We perform the first reduction using a block-size $\beta = c_\beta(\log|\Delta_\mathbf{K}|)^k$. It still costs $L_{|\Delta_\mathbf{K}|}\big(k, o(1)\big)$ and gives rise to an algebraic integer x_0 and an integral ideal $\mathfrak{a}^{(0)}$ such that $\langle x_0\rangle = \mathfrak{a}\cdot\mathfrak{a}^{(0)}$.

According to Corollary 1.4.7, the norm of $\mathfrak{a}^{(0)}$ satisfies

$$\mathcal{N}\left(\mathfrak{a}^{(0)}\right) \leq L_{|\Delta_{\mathbf{K}}|}\left(2\alpha - k, \frac{n_0^2}{2c_\beta}\right) = L_{|\Delta_{\mathbf{K}}|}\left(3k, \frac{n_0^2}{2c_\beta}\right).$$

We make use of the same randomization process as in the medium-case and obtain a $L_{|\Delta_{\mathbf{K}}|}(2k, s_0)$-smooth ideal in time $L_{|\Delta_{\mathbf{K}}|}\left(k, \left(\frac{9k^2 n_0^2}{4c_\beta}\right)^{\frac{1}{3}}\right)$, for $s_0^3 = \frac{kn_0^4}{2c_\beta^2}$ chosen to minimize this cost.

The subsequent steps begin with an ideal of norm less than $L_{|\Delta_{\mathbf{K}}|}\left(k\left(1+\frac{1}{2^i}\right), s_i\right)$. Then, by fixing $\delta = k\left(1+\frac{1}{2^{i+1}}\right)$, we obtain an ideal $\mathfrak{a}^{(i+1)}$ such that is norm is upper-bounded by

$$L_{|\Delta_{\mathbf{K}}|}\left(k\left(2+\frac{1}{2^{i+1}}\right), \frac{n_0 s_i}{c_d}\right).$$

so that, assuming Heuristic 1.4.20, we can find an ideal that is $L_{|\Delta_{\mathbf{K}}|}\left(k\left(1+\frac{1}{2^{i+1}}\right), s_{i+1}\right)$-smooth in time

$$L_{|\Delta_{\mathbf{K}}|}\left(k, \frac{kn_0 s_i}{c_d s_{i+1}}\right).$$

In the same way, setting $l = \left\lceil \log_2\left(\frac{1}{k}\log\left(\frac{\log|\Delta_{\mathbf{K}}|}{\log\log|\Delta_{\mathbf{K}}|}\right)\right)\right\rceil$ implies that after step l, the ideals involved are $L_{|\Delta_{\mathbf{K}}|}(k, e\cdot s_l)$-smooth; here we want $e\cdot s_l$ to be smaller than c_b.

Let $y>0$ be a constant such that, at each step, the runtime of the smoothness tests is below $L_{|\Delta_{\mathbf{K}}|}(k, y)$. That means that for all i, it is the case that $\frac{kn_0 s_i}{c_d s_{i+1}} \leq y$. Then, by fixing $c_d = \frac{kn_0}{y}\cdot\left(\frac{es_0}{c_b}\right)^{\frac{1}{l}}$ and $s_{i+1} = s_i \cdot \left(\frac{c_b}{es_0}\right)^{\frac{1}{l}}$, the previous equation is satisfied, resulting in

$$s_l = s_0 \cdot \left(\frac{c_b}{es_0}\right) \qquad \Longleftrightarrow \qquad e\cdot s_l = c_b.$$

As $y>0$ can be chosen arbitrarily small, each step has a runtime in $L_{|\Delta_{\mathbf{K}}|}(k, o(1))$ and the initial-reduction cost can also be chosen that small, for c_β sufficiently large. The remaining part consisting in solving the linear system works in the same way as for the previous case and we can conclude that the complexity of our algorithm for solving the PIP is the same as the complexity of the class group computation. Again, Remark 5.5.3 holds so that we can reduce the complexity to

$$L_{|\Delta_{\mathbf{K}}|}\left(k, \left(\frac{k^2 n_0 \omega^2}{2(\omega-1)}\right)^{\frac{1}{2}}\right).$$

5.5.3 The small-degree case

Again, we only give a brief summary of the descent. Here we have $2\alpha < \gamma$ and k denotes $\frac{\gamma}{2}$.

The initial BKZ-reduction provides an ideal of norm bounded by $L_{|\Delta_\mathbf{K}|}\left(\alpha+\gamma, n_0 d_0\right)$ in time $L_{|\Delta_\mathbf{K}|}\left(k, o(1)\right)$. We can find an ideal that is $L_{|\Delta_\mathbf{K}|}\left(k+\alpha, s_0\right)$-smooth in time $L_{|\Delta_\mathbf{K}|}\left(k, \frac{k n_0 d_0}{s_0}\right)$ as the cost of a single application of ECM is negligible — because $\frac{k+\alpha}{2} < k$.

Then, every subsequent step takes as input an ideal of norm less than $L_{|\Delta_\mathbf{K}|}\left(k+\frac{\alpha}{2^i}, s_i\right)$. Then, looking for a small vector in the sublattice of dimension $d = c_d\left(\frac{\log|\Delta_\mathbf{K}|}{\log\log|\Delta_\mathbf{K}|}\right)^{\frac{\alpha}{2^{i+1}}}$ leads to a new ideal of norm upper bounded by $L_{|\Delta_\mathbf{K}|}\left(2k+\frac{\alpha}{2^{i+1}}, d_0 c_d\right)$. Again we expect, assuming Heuristic 1.4.20, to find one that is $L_{|\Delta_\mathbf{K}|}\left(k+\frac{\alpha}{2^{i+1}}, s_{i+1}\right)$-smooth in time $L_{|\Delta_\mathbf{K}|}\left(k, \frac{k d_0 c_d}{s_{i+1}}\right)$.

At final step $l = \left\lceil \log_2\left(\frac{1}{\alpha}\log\left(\frac{\log|\Delta_\mathbf{K}|}{\log\log|\Delta_\mathbf{K}|}\right)\right)\right\rceil$, we have ideals that are $L_{|\Delta_\mathbf{K}|}(k, e\cdot s_l)$-smooth and we want $e\cdot s_l$ to be smaller than $c_b = \left(\frac{k d_0 c_t}{\omega}\right)^{\frac{1}{2}}$. Note that, at this point, $d = c_d e\left(1+o(1)\right)$ for the same reason as above. Hence c_d may be as small as $\frac{1}{e}$ and the cost of the final smoothness test is lower-bounded by

$$L_{|\Delta_\mathbf{K}|}\left(k, \frac{k d_0}{e\cdot s_l}\right) \geq L_{|\Delta_\mathbf{K}|}\left(k, \frac{k d_0}{c_b}\right) = L_{|\Delta_\mathbf{K}|}\left(k, \left(\frac{k d_0 \omega}{c_t}\right)^{\frac{1}{2}}\right).$$

This last smoothness test dominates the overall complexity of the descent phase, as we can always choose c_d and x_i such that the runtimes of the other smoothness tests become arbitrarily small. In addition, this part is dominated by the class group computation: indeed, $\left(\frac{k d_0 \omega}{c_t}\right)^{\frac{1}{2}} \leq (\omega+1)\left(\frac{k d_0 c_t}{\omega}\right)^{\frac{1}{2}}$ because $c_t \geq 1 > \frac{w}{w+1}$. Again, we can improve this algorithm as explained in Remark 5.5.3 and finally get a complexity of

$$L_{|\Delta_\mathbf{K}|}\left(k, \left(\frac{k d_0 c_t \omega^2}{\omega-1}\right)^{\frac{1}{2}}\right).$$

Part III

Applications to Cryptology

Chapter 6

PIP solution in cyclotomic fields and cryptanalysis of an FHE scheme

Contents

Homomorphic encryption is a form of encryption that allows any sequence of operations to be carried out on a ciphertext, thus generating an encrypted result which, when decrypted, matches the result of the same sequence of operations performed on the plaintext. Gentry [Gen09], using lattice-based cryptography, described the first construction for a Fully Homomorphic Encryption (FHE) scheme. Gentry's scheme supports both addition and multiplication operations on ciphertexts, from which it is possible to construct circuits for performing arbitrary computations.

6.1 Situation of the problem and cryptosystems that rely on SPIP

Among all the FHE schemes proposed in the last decade, the security of a couple of them directly collapses if is possible to find relatively short generators in principal ideals. This is the case of the proposal of Smart and Vercauteren [SV10], which is a simplified version of the original scheme of Gentry [Gen09]. Other schemes based on the same security assumption include the Soliloquy scheme of Campbell, Groves, and Shepherd [CGS14] and the candidates for multilinear maps [GGH13, LSS14]. More formally, the underlying — presumably hard — problem is the following one, already known as SPIP (Short Principal Ideal Problem) or SG-PIP (Short Generator-Principal Ideal Problem): given some $\mathbf{Z}$-basis of a principal ideal with a promise that it possesses a "short" generator g for the Euclidean norm, find this generator or at least a short enough generator of this ideal.

The strategy to address this problem roughly splits into two main steps:

1. **PIP:** given the $\mathbf{Z}$-basis of the ideal, find a generator, not necessarily short, that is $g' = g \cdot u$ for a unit u.

2. **Reduction:** from g', find a short generator of the ideal.

Recently, several results have allowed to deal with the second step. Indeed, Campbell, Groves, and Shepherd [CGS14] claimed in 2014 an — although unproven — efficient solution for power-of-two cyclotomic fields, confirmed by experiments conducted by Schanck [Sch15] in 2015. Eventually, the proof was provided by Cramer, Ducas, Peikert, and Regev [CDPR16], together with an extension to all prime-power cyclotomic fields. Throughout this chapter, we focus on the resolution of the first step, the Principal Ideal Problem (PIP). Nonetheless, for completeness, we present briefly the reduction from SPIP to PIP in Section 6.3.4.

As a direct illustration of the resolution of this problem, we present an attack on the scheme that Smart and Vercauteren present in [SV10], which leads to a *full key recovery*. This attack is our key thread throughout the exposition of the algorithm. Before going any further in the details of the attack, we recall in Algorithm 5 the key generation process in the case

of power-of-two cyclotomic fields. This instantiation is the one chosen by the authors for presenting their implementation results.

Algorithm 5 Key Generation of the scheme [SV10].

Input: The security parameter $n = 2^m$.
Output: A pair $(\mathsf{sk}, \mathsf{pk})$ of secret/public keys.

1: Set $\Phi_{2n}(X) = X^n + 1$ as the polynomial defining the cyclotomic field $\mathbf{K} = \mathbf{Q}(\zeta_{2n})$
2: **repeat**
3: $\quad G(X) = 1 + 2 \cdot S(X)$ for $S(X)$ of degree $n-1$ with coefficients absolutely bounded by $2^{\sqrt{n}}$
4: **until** the norm $\mathcal{N}\big(\langle G(\zeta_{2n})\rangle\big)$ is prime
5: Set $g = G(\zeta_{2n}) \in \mathcal{O}_{\mathbf{K}}$
6: **return** $\big(\mathsf{sk} = g, \mathsf{pk} = \mathrm{HNF}\big(\langle g \rangle\big)\big)$

Remark 6.1.1. The public key can be any $\mathbf{Z}$-basis of the ideal generated by g, or even a two-elements representation of this ideal. Indeed, [SV10] provides the public key as a pair of elements that generates the lattice.

As our attack consists in a full secret-key recovery, based on just the public key, the encryption and decryption procedures are irrelevant for our purposes and thus omitted. Even though this work is more concerned with the principal ideal problem rather than the reduction step, we emphasize the fact that the short generator resulting from this reduction can be any of the $g \cdot \zeta_{2n}^i$, for $1 \leq i \leq 2n$, which all have the same Euclidean norm. This is not an issue, since all these keys are equivalent with regard to the decryption procedure. In addition, in this precise construction of the Smart and Vercauteren FHE scheme, the only odd coefficient of $G(X)$ is the constant one, so that we may recover the exact generator g readily — a multiplication by ζ_{2n} is only a shift in the coefficients.

The overall complexity of our attack is subexponential, namely $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)$. This beats the previous state of the art which runs in time $L_{|\Delta_{\mathbf{K}}|}\left(\frac{2}{3} + \varepsilon\right)$, derived from the combined work of [BF14] and [CDPR16]. We also provide an implementation that breaks the smallest instantiations of the scheme [SV10] described in the original paper. Most of the tools have already been explained in Chapter 5, since this attack predates the generalization presented in Chapter 5. Solving the PIP is also the subject of research on quantum algorithms: Biasse and Song [BS16] described a quantum polynomial-time solution for classes of number fields of arbitrary degree.

We have already computed the discriminant of $\mathbf{Q}(\zeta_m)$ in Section 4.1. For a prime-power cyclotomic field, we get $\left|\Delta_{\mathbf{Q}(\zeta_{p^k})}\right| = p^{(kp-k-1)p^{k-1}}$. In particular, when $p = 2$, it is the case that $\left|\Delta_{\mathbf{Q}(\zeta_{2^{m+1}})}\right| = 2^{m2^m}$. For power-of-two cyclotomic fields, we then have $L_{|\Delta_{\mathbf{K}}|}(\alpha) = 2^{O(n^\alpha \log(n))}$. Thus, writing the complexity as $L_{|\Delta_{\mathbf{K}}|}(\alpha)$ or $2^{O(n^\alpha \log(n))}$ is equivalent. We choose to use the L-notation, since it facilitates the exposition of the complexities presented in this chapter.

6.2 Solving the PIP or how to perform a full key recovery?

We recall that our ultimate goal is to perform a full key recovery given only the public elements. As mentioned in [SV10], this problem is obviously much more difficult than recovering a plain-text from a cipher-text which is based on the bounded distance decoding problem and the security level is set according to the alleged hardness of this problem. We first give an overview of the whole strategy and then present each part in-depth. But before going any further into the details of the attack, let us fix the notation. The number field where the PIP is defined is $\mathbf{Q}(\zeta_{2n})$, for $n = 2^m$, defined by the polynomial $X^n + 1$, as in Section 6.1. For simplicity we write ζ for ζ_{2n}. Though we focus on power-of-two cyclotomic fields, all our results can be easily generalized to arbitrary prime-power cyclotomic fields. Our starting point is the public key, that is a somewhat "bad" basis of the principal ideal $\mathfrak{a} = \langle g \rangle$, generated by the secret key g.

Before any other operations, the dimension of the ideals involved is shrunk by half by reducing the problem to an equivalent one in the *totally real subfield* $\mathbf{Q}\left(\zeta + \zeta^{-1}\right)$. This is not mandatory, but it makes the computation easier. This part of the algorithm is a straightforward consequence of the Gentry-Szydlo algorithm introduced in [GS02]. The problem is now reduced to the search for a generator of an ideal $\mathfrak{a}^+$ in the totally real subfield. The strategy is then to first iteratively reduce the size of the ideals involved, until we eventually reach a B-smooth ideal $\mathfrak{a}^s$, for a fixed bound $B > 0$, and an algebraic integer h such that $\langle h \rangle = \mathfrak{a}^+ \cdot \mathfrak{a}^s$. This is the *descent phase*.

The next step is finding a generator of $\mathfrak{a}^s$. We use a strategy based on class group computation. It consists in finding a generating set of all the relations among the generators of the class group, and then rewrite the input ideal with respect to these generators. Then we can recover a generator h_0 of $\mathfrak{a}^s$ by solving a linear system of equations. The latter allows us to derive a generator of the ideal $\mathfrak{a}^+$: $h \cdot h_0^{-1}$. A generator of the public-key ideal is then obtained by lifting a generator of the ideal $\mathfrak{a}^+$ from the totally real subfield to the initial number field $\mathbf{Q}(\zeta)$; for this step it suffices to multiply the current generator by another algebraic integer obtained during the computation. Finally, given a solution to the PIP, it remains to recover the secret key, which is done by reducing the generator found to a short one, as described in [CDPR16].

Consequently, the full algorithm can be split in four main steps, which are, in a nutshell:

1. Perform a reduction from the cyclotomic field to its totally real subfield, allowing to work in smaller dimension.
2. Then a descent lowers the size of the ideals involved.
3. Collect relations and run linear algebra to construct small ideals and a generator.
4. Eventually derive a small generator from the bigger one.

6.3 Description of the algorithm

Let us now get into the details of all steps sketched above.

6.3.1 Step 1: Reduction to the totally real subfield

We may assume that the input public key is specified as a $\mathbf{Z}$-basis $(b_1, \ldots, b_n)$ of an ideal $\mathfrak{a}$ belonging to the cyclotomic field $\mathbf{Q}(\zeta)$ of dimension[1] n. The larger the dimension is, the harder the algebraic numbers are to handle and even only to represent. However, it is possible to halve the dimension. The main part of this step relies on the so-called *Gentry-Szydlo* algorithm, first described in [GS02] as an attack against the NTRU scheme and later revised and generalized by Lenstra and Silverberg in [LS14].

This original algorithm from [GS02] takes as input a $\mathbf{Z}$-basis of an ideal $\mathfrak{a}$ in the ring $\mathbf{Z}[X]/\langle X^n+1\rangle$ — with the promise to be principal — and the algebraic integer $u\cdot\bar{u}$, for u a generator of $\mathfrak{a}$. Here, $\bar{u}$ denotes the conjugate of u for the automorphism defined by $\zeta \mapsto \zeta^{-1}$. It then recovers in polynomial time the element u. In our case, we can not perform the recovery of the generator g, the secret key of the scheme, since *a priori* we do not have access to any kind of information about the product $g\cdot\bar{g}$.

To overcome this difficulty, we introduce another algebraic integer $u = \mathcal{N}(g)\, g\, \bar{g}^{-1}$, as described by Garg, Gentry, and Halevi in [GGH13, Section 7.8.1]. Here the norm factor is included only to avoid the introduction of denominators in the definition of u. Although u is still unknown at this point, thanks to the $\mathbf{Z}$-basis of $\langle g\rangle$ we can construct a $\mathbf{Z}$-basis of $\langle u\rangle$ and compute the product $u\cdot\bar{u}$, as in that case, it simply corresponds to $\mathcal{N}(g)^2$.

Hence, we are able to compute u in polynomial time using the Gentry-Szydlo algorithm, and from this element u, we can directly reconstruct $g\,\bar{g}^{-1}$. Using the basis of $\mathfrak{a}$, we then introduce the family of vectors

$$c_i = b_i\left(1+\frac{\bar{g}}{g}\right),$$

providing a basis of the ideal $\mathfrak{a}^+$ generated by $g+\bar{g}$. This ideal belongs to the totally real subfield $\mathbf{Q}\left(\zeta+\zeta^{-1}\right)$, of index 2 in $\mathbf{Q}(\zeta)$. From now on, we denote by $\mathcal{O}_{\mathbf{K}}^+$ the ring of integers of $\mathbf{Q}\left(\zeta+\zeta^{-1}\right)$, corresponding to $\mathcal{O}_{\mathbf{K}}\cap\mathbf{Q}\left(\zeta+\zeta^{-1}\right)$.

Let us suppose briefly that we know the generator $g+\bar{g}$ of $\mathfrak{a}^+$. Then it would be sufficient to multiply it by $\frac{1}{1+g\,\bar{g}^{-1}}$ to recover the secret key g. Hence, we have reduced the problem of finding a generator of the ideal $\mathfrak{a}$ belonging to the cyclotomic field of dimension n to the one of finding a generator of ideal $\mathfrak{a}^+$ that belongs to the totally real subfield, whose dimension is $\frac{n}{2}$. For a more detailed presentation of this technique, see [GGH13, Theorem 8].

[1] The smallest security parameters of the Smart and Vercauteren scheme is $n = 256$.

Note that even though the generator is known up to a unit — i.e. $(g+\bar{g})\cdot v$ for $v \in \mathcal{U}_{\mathbf{Q}(\zeta)}$ — the generator of $\mathfrak{a}$ that is found is $g \cdot v$. Due to the last reduction part, this suffices to find a short generator.

One could wonder if working in a real field is required for the remaining parts of the attack. This is not the case: all we are interested in is halving the dimension. For the asymptotic complexity, this initial step is mostly irrelevant since it only gives a speedup of a constant factor in the exponent. But in practice, it allows to double the dimension of the tractable cases, implying attacking security parameters that are twice bigger! See also the final paragraph of Section 6.5.

6.3.2 Step 2: Descent phase

Let us momentarily set aside the algebraic integer obtained in the previous phase and only focus on the ideal $\mathfrak{a}^+$. By construction, it is principal and generated by $g+\bar{g}$. In the sequel, all the computations are performed in the totally real subfield of dimension $\frac{n}{2}$.

The goal of this phase is to find an algebraic integer h and a B-smooth principal ideal $\mathfrak{a}^s$, such that $\langle h \rangle = \mathfrak{a}^+ \cdot \mathfrak{a}^s$, for a certain bound $B > 0$. These objects are discovered iteratively, by generating at each step ideals of norm smaller and smaller. Since we want a global complexity $L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}\right)$, the smoothness bound B is chosen as $L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}\right)$. In order to bootstrap this descent, we first need to find an ideal that splits as a product of multiple prime ideals of controlled norm, that is in our case, upper bounded by $L_{|\Delta_\mathbf{K}|}(1)$. The method shares many similarities with the one described in Section 5.5. However, the special case of cyclotomic fields allows a number of simplifications. For the sake of simplicity, we omit the second constant of the complexity throughout the description of the algorithm. The method described in Section 5.5 is used to provide an estimate in Section 6.4.

Initial round. As mentioned, we aim to construct efficiently an $L_{|\Delta_\mathbf{K}|}(1)$-smooth principal ideal from $\mathfrak{a}^+$. Formally, we want to prove the following:

Theorem 6.3.1. *Let $\mathbf{K}$ be a number field. Assuming Heuristic 1.4.20, from any ideal $\mathfrak{a} \subset \mathcal{O}_\mathbf{K}$, it is possible to generate in expected time $L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}\right)$ an integral ideal $\mathfrak{b}$ that is $L_{|\Delta_\mathbf{K}|}(1)$-smooth and an integer v such that*

$$\langle v \rangle = \mathfrak{a} \cdot \mathfrak{b}.$$

The difficulty of this preliminary part is that *a priori* the norm of the input ideal $\mathfrak{a}$ can be large. We thus want to construct at first an ideal whose norm is bounded independently from $\mathcal{N}(\mathfrak{a})$ and that belongs to the same ideal class as $\mathfrak{a}$. We proceed by ideal-lattice reduction. Using the canonical embedding σ, any integral ideal $\mathfrak{a}$ can be viewed as a lattice: $\sigma(\mathfrak{a})$. As

usual when dealing with lattice reduction, we are interested in small vectors, or in the present case equivalently, algebraic integers with small norm.

From a similar analysis as the one presented in Section 4.3, it follows from Theorem 1.1.10 and Lemma 1.4.9 that the smallest vector v of the BKZ$_\beta$-reduced basis of $\sigma(\mathfrak{a})$ has a norm that satisfies $\|\sigma(v)\| \leq \beta^{\frac{n-1}{2(\beta-1)}} \mathcal{N}(\mathfrak{a})^{\frac{1}{n}} |\Delta_{\mathbf{K}}|^{\frac{1}{2n}}$. The cost of this reduction is upper bounded by $\mathrm{Poly}\left(n, \log \mathcal{N}(\mathfrak{a})\right) \cdot 2^{O(\beta)}$. Since the ideal $\mathfrak{a}$ contains $\langle v \rangle$, there exists a unique integral ideal $\mathfrak{b}$ satisfying $\langle v \rangle = \mathfrak{a} \cdot \mathfrak{b}$. From the upper bound on $\|\sigma(v)\|$, we can bound the norm of this new ideal $\mathfrak{b}$.

From Lemma 1.4.2, the field norm of v is upper bounded by the n-th power of this bound and so $\mathcal{N}(\langle v \rangle) \leq \beta^{\frac{n(n-1)}{2(\beta-1)}} \cdot \sqrt{|\Delta_{\mathbf{K}}|} \cdot \mathcal{N}(\mathfrak{a})$. By the multiplicative property of the norm, this inequality results in

$$\mathcal{N}(\mathfrak{b}) \leq \beta^{\frac{n(n-1)}{2(\beta-1)}} \cdot \sqrt{|\Delta_{\mathbf{K}}|}.$$

Because $\mathbf{K}$ is a cyclotomic field, we are able to choose a block-size $\beta = \log Ł_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)$ since $\log Ł_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right) = n^{1/2+o(1)} \leq n$. Then we are able to generate in time $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)$ an integral ideal of norm bounded by $L_{|\Delta_{\mathbf{K}}|}\left(\frac{3}{2}\right)$.

This last result allows us to find an ideal of norm bounded independently from $\mathcal{N}(\mathfrak{a})$. We then want this new ideal to split as a product of multiple prime ideals of controlled norms. Thanks to Corollary 1.4.21, the probability for an integral ideal $\mathfrak{b}$ of norm bounded by $L_{|\Delta_{\mathbf{K}}|}\left(\frac{3}{2}\right)$ to be $L_{|\Delta_{\mathbf{K}}|}(1)$-smooth is greater than $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)^{-1}$. In addition, using ECM to test for smoothness keeps the complexity in $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)$ (see Proposition 1.4.18). Therefore, repeating the last construction $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)$ times on randomized independent inputs eventually yields an $L_{|\Delta_{\mathbf{K}}|}(1)$-smooth ideal. The simplest strategy to perform this randomization of the input ideal is to compose it with some factors of norm less than $B = Ł_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)$. Formally, we denote by $\mathcal{B} = \{\mathfrak{p}_1, \ldots, \mathfrak{p}_{|\mathcal{B}|}\}$ the set of all prime ideals of norm upper bounded by $Ł_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)$. Let $k, A > 0$ be fixed integers. We choose $\mathfrak{p}_{j_1}, \ldots, \mathfrak{p}_{j_k}$ prime ideals of norm below $Ł_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)$. Then for any k-tuple $(e_1, \ldots, e_k) \in \{1, \ldots, A\}^k$, we have

$$\mathcal{N}\left(\mathfrak{a} \cdot \prod_{i=1}^{k} \mathfrak{p}_{j_i}^{e_i}\right) \leq \mathcal{N}(\mathfrak{a}) \cdot \prod_{i=1}^{k} \mathcal{N}\left(\mathfrak{p}_{j_i}^{e_i}\right) \leq \mathcal{N}(\mathfrak{a}) \cdot L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)^{k \cdot A} = \mathcal{N}(\mathfrak{a}) \cdot L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right).$$

Thus, the randomization can be done by choosing uniformly at random the tuple $(e_1, \ldots, e_k)$ and k prime ideals in $\mathcal{B}$. Since from Landau's Prime Ideal theorem, $|\mathcal{B}| = L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)$, the set of possible samples is large enough for our purposes.

Other ways to perform the randomization may be by randomizing directly the lattice reduction algorithm or by enumerating points of the lattice of norm close to the norm guarantee and change the basis vectors by freshly enumerated ones. The latter would be useful in practice as it reduces the number of reductions (see Remark 4.2.3 for more details).

This last remark concludes the proof of Theorem 6.3.1. The full outline of this bootstrap approach is given in Algorithm 6.

Algorithm 6 Initial reduction to an $L_{|\Delta_{\mathbf{K}}|}(1)$-smooth ideal.

Input: An ideal $\mathfrak{a}$ of arbitrarily large norm and $\mathcal{B}$, A, and k as above for the randomization.
Output: An integer v and an $L_{|\Delta_{\mathbf{K}}|}(1)$-smooth ideal $\mathfrak{b}$ such that $\langle v \rangle = \mathfrak{a}\mathfrak{b}$.

1: Set $\mathfrak{b} = \mathfrak{a}$
2: **while** $\mathfrak{b}$ is not $L_{|\Delta_{\mathbf{K}}|}(1)$-smooth **do**
3: Choose $\mathfrak{p}_{j_1}, \cdots, \mathfrak{p}_{j_k}$ uniformly at random in $\mathcal{B}$
4: Set $\tilde{\mathfrak{a}} = \mathfrak{a} \cdot \prod \mathfrak{p}_{j_i}^{e_i}$ for random $e_i \in \{-A, \ldots, A\}$
5: Generate v and $\tilde{\mathfrak{b}}$ from $\tilde{\mathfrak{a}}$ as in Theorem 6.3.1
6: Set $\mathfrak{b} = \tilde{\mathfrak{b}} \cdot \prod \mathfrak{p}_{j_i}^{-e_i}$
7: **end while**
8: **return** v and $\mathfrak{b}$

Subsequent steps. In the proof of Theorem 6.3.1, we use a *classical* reduction in order to find a short element in the embedding of the considered ideal. We could not use directly Cheon's trick here since the norm of the ideal $\mathfrak{a}^+$ — and so the determinant of its coefficient embedding — is potentially large. Nonetheless, the norm of prime ideals appearing in the factorization are by construction bounded, hence a natural question is to look at the bounds that apply when applying the method referred to below. The systematic treatment of this question is the aim of Theorem 6.3.2.

Theorem 6.3.2. *Let $\mathfrak{a}$ be an integral ideal of norm below $L_{|\Delta_{\mathbf{K}}|}(\alpha)$, for $\frac{1}{2} \leq \alpha \leq 1$. Then, in expected time $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)$, it is possible to construct an algebraic integer v and an $L_{|\Delta_{\mathbf{K}}|}\left(\frac{2\alpha+1}{4}\right)$-smooth ideal $\mathfrak{b}$ such that*

$$\langle v \rangle = \mathfrak{a} \cdot \mathfrak{b}.$$

Proof. The core of the proof is similar to the proof of Theorem 6.3.1 as it heavily relies on lattice reduction and randomization techniques. The major difference lies in the embedding with respect to which the reduction is performed. In Theorem 6.3.1, the canonical embedding is used, whereas here we use the coefficient embedding ς. It avoids the possibility that a power of the discriminant occurs in the field norm of the output of BKZ. Nonetheless, remark that since we work in the totally real subfield, we cannot use a naive coefficients embedding of this subfield. In order to benefit from the nice shape of the defining polynomial $X^N + 1$ of the cyclotomic field, we use instead a fold-in-two strategy: the embedding of $\mathcal{O}_{\mathbf{K}}^+$ is defined as the coefficient embedding ς^+ for the $\mathbf{Z}$-basis $(\zeta^i + \zeta^{-i})_i$. Hence, for any $v \in \mathcal{O}_{\mathbf{K}}^+$,

$$\|\varsigma(v)\| = \sqrt{2}\|\varsigma^+(v)\|.$$

Let $\mathcal{L} = \varsigma^+(\mathfrak{a})$ be the embedding of $\mathfrak{a}$. Its determinant is $\mathcal{N}(\mathfrak{a})$. Then, with the same block-size $\beta = \log L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}\right) = O\left(\sqrt{n}\log n\right)$, we have

$$\det \mathcal{L} \leq L_{|\Delta_\mathbf{K}|}(\alpha) = 2^{O(n^\alpha \log(n))} \leq \beta^{\frac{n^2}{2\beta}}.$$

Using the algorithm of Theorem 1.1.10 yields in time $L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}\right)$ an integer v satisfying

$$\|\varsigma^+(v)\| \leq \beta^{\sqrt{\frac{2\log_\beta(\det\mathcal{L})}{\beta}}(1+o(1))} \leq L_{|\Delta_\mathbf{K}|}\left(\frac{\alpha}{2} - \frac{1}{4}\right).$$

Using Corollary 1.4.7 to formulate this in terms of the field norm induces

$$\mathcal{N}_{\mathbf{K}^+/\mathbf{Q}}(v) \leq \left(\sqrt{2}(n+1)\right)^n \cdot \|\varsigma(v)\|^n = L_{|\Delta_\mathbf{K}|}(1) \cdot L_{|\Delta_\mathbf{K}|}\left(\frac{\alpha}{2} + \frac{3}{4}\right).$$

Since $\alpha \geq \frac{1}{2}$, we then have $\mathcal{N}(\langle v\rangle) = \mathcal{N}_{\mathbf{K}/\mathbf{Q}}(v) \leq L_{|\Delta_\mathbf{K}|}\left(\frac{\alpha}{2} + \frac{3}{4}\right)$.

Because the ideal $\mathfrak{a}$ contains $\langle v\rangle$, there exists a unique integral ideal $\mathfrak{b}$, satisfying $\langle v\rangle = \mathfrak{a}\cdot\mathfrak{b}$. We find that $\mathcal{N}(\mathfrak{b}) \leq L_{|\Delta_\mathbf{K}|}\left(\frac{\alpha}{2} + \frac{3}{4}\right)$ from the multiplicative property of the norm and $\mathcal{N}(\mathfrak{a}) = L_{|\Delta_\mathbf{K}|}(1) \leq L_{|\Delta_\mathbf{K}|}\left(\frac{\alpha}{2} + \frac{3}{4}\right)$. Under Heuristic 1.4.20, this ideal is $L_{|\Delta_\mathbf{K}|}\left(\frac{\alpha}{2} + \frac{1}{4}\right)$-smooth with probability $L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}\right)$. Eventually performing the randomization-and-repeat technique as in the initial round, this reduction of the coefficient embedding yields the desired couple $(v, \mathfrak{b})$ in expected time $L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}\right)$. □

Descending to B-smoothness. After the first round, we end up with an $L_{|\Delta_\mathbf{K}|}(1)$-smooth ideal, denoted by $\mathfrak{a}^{(0)}$, and an algebraic integer $h^{(0)}$ satisfying

$$\left\langle h^{(0)}\right\rangle = \mathfrak{a}^+ \cdot \mathfrak{a}^{(0)},$$

with $\mathfrak{a}^+$ the ideal of the totally real subfield obtained as a result of the Gentry-Szydlo algorithm. The factorization of $\mathfrak{a}^{(0)}$ gives

$$\mathfrak{a}^{(0)} = \prod_j \mathfrak{a}_j^{(0)},$$

where the $\mathfrak{a}_j^{(0)}$ are integral prime ideals of norm upper bounded by $L_{|\Delta_\mathbf{K}|}(1)$. Taking the norms of the ideals involved in this equality, it is found that the number of terms in this product is $O(n_\mathcal{I})$, with $n_\mathcal{I} = \left(\frac{\log|\Delta_\mathbf{K}|}{\log\log|\Delta_\mathbf{K}|}\right)^{\frac{1}{2}} = O\left(\sqrt{n}\right)$. Then applying Theorem 6.3.2 to each small ideal $\mathfrak{a}_j^{(0)}$ gives rise in expected time $L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}\right)$ to ideals $\mathfrak{a}_j^{(1)}$ that are $L_{|\Delta_\mathbf{K}|}\left(\frac{2\times 1+1}{4}\right) = L_{|\Delta_\mathbf{K}|}\left(\frac{3}{4}\right)$-smooth and integers $h_j^{(1)}$ such that for every j,

$$\left\langle h_j^{(1)}\right\rangle = \mathfrak{a}_j^{(0)} \cdot \mathfrak{a}_j^{(1)}.$$

For each factor $\mathfrak{a}_j^{(1)}$, let us write its prime decomposition:

$$\mathfrak{a}_j^{(1)} = \prod_k \mathfrak{a}_{j,k}^{(1)}.$$

Once again, the number of terms appearing is $O(n_{\mathcal{I}})$. Since we have $\mathcal{N}\left(\mathfrak{a}_{j,k}^{(1)}\right) \leq L_{|\Delta_{\mathbf{K}}|}\left(\frac{3}{4}\right)$, performing the same procedure for each ideal $\mathfrak{a}_{j,k}^{(1)}$ then yields $L_{|\Delta_{\mathbf{K}}|}\left(\frac{5}{8}\right)$-smooth ideals $\mathfrak{a}_{j,k}^{(2)}$ and integers $h_{j,k}^{(2)}$ such that

$$\left\langle h_{j,k}^{(2)} \right\rangle = \mathfrak{a}_{j,k}^{(1)} \cdot \mathfrak{a}_{j,k}^{(2)},$$

once again in expected time $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)$. Remark that this smoothness bound $L_{|\Delta_{\mathbf{K}}|}\left(\frac{5}{8}\right)$ is obtained as $L_{|\Delta_{\mathbf{K}}|}\left(\frac{2\times 3+1}{4}\right)$, as follows from Theorem 6.3.2. This reasoning naturally leads to an iterative strategy for reduction. At step k, we want to reduce an ideal $\mathfrak{a}_{a_1,\ldots,a_{k-1}}^{(k-1)}$ which is $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}+\frac{1}{2^{k+1}}\right)$-smooth. As before, we have a decomposition — with $O(n_{\mathcal{I}})$ terms — in smaller ideals:

$$\mathfrak{a}_{a_1,\ldots,a_{k-1}}^{(k-1)} = \prod_j \mathfrak{a}_{a_1,\ldots,a_{k-1},j}^{(k-1)}.$$

Using Theorem 6.3.2 for each factor $\mathfrak{a}_{a_1,\ldots,a_{k-1},j}^{(k-1)}$ of norm bounded by $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}+\frac{1}{2^{k+1}}\right)$ leads to $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}+\frac{1}{2^{k+2}}\right)$-smooth ideals $\mathfrak{a}_{a_1,\ldots,a_{k-1},j}^{(k)}$ and algebraic integers $h_{a_1,\ldots,a_{k-1},j}^{(k)}$ such that

$$\left\langle h_{a_1,\ldots,a_{k-1},j}^{(k)} \right\rangle = \mathfrak{a}_{a_1,\ldots,a_{k-1},j}^{(k-1)} \cdot \mathfrak{a}_{a_1,\ldots,a_{k-1},j}^{(k)},$$

since $\frac{2\times\left(\frac{1}{2}+\frac{1}{2^{k+1}}\right)+1}{4} = \frac{1}{2}+\frac{1}{2^{k+2}}$.

As a consequence, one can generate $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}+\frac{1}{\log n}\right)$-smooth ideals with the previous method in at most $\lceil \log_2 \log n \rceil$ steps. At this point, only $(n_{\mathcal{I}})^{\lceil \log_2(\log n)\rceil}$ ideals and algebraic integers appear since at each step this number is multiplied by a factor $O(n_{\mathcal{I}})$. As deriving a single integer/ideal pair requires expected time $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)$, the overall complexity remains $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)$.

The final step is done in the same way as in Section 5.5. However, as $|\Delta_{\mathbf{K}}| = n^n$, a quick calculation reveals that

$$\begin{aligned}\log L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}+\frac{1}{\log n}\right) &= O\left(n^{\frac{1}{2}+\frac{1}{\log n}}\log(n)\right)\\ &= O\left(n^{\frac{1}{2}}\log(n)\right)\cdot n^{\frac{1}{\log n}}.\end{aligned}$$

Since the last factor $n^{\frac{1}{\log n}}$ is $e = \exp(1)$, we obtain that

$$\log L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}+\frac{1}{\log n}\right) = \log L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right),$$

so that after at most $\lceil \log_2 \log n \rceil$ steps, we have ideals that are $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)$-smooth.

At the end of this final round, we may express the input ideal as the product of ideals for which we know a generator and others that have by construction norms bounded by $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)$. Let l denote the index of the final step. To avoid having to deal with inverse ideals, we may assume without loss of generality[2] that l is even. Explicitly, we have

$$\begin{aligned}
\langle h^{(0)} \rangle = \mathfrak{a}^+ \cdot \mathfrak{a}^{(0)} &= \mathfrak{a}^+ \cdot \prod_{a_1} \mathfrak{a}_{a_1}^{(0)} \\
&= \mathfrak{a}^+ \cdot \left\langle \frac{\prod_{a_1} h_{a_1}^{(1)} \prod_{a_1,a_2,a_3} h_{a_1,a_2,a_3}^{(3)}}{\prod_{a_1,a_2} h_{a_1,a_2}^{(2)}} \right\rangle \cdot \prod_{a_1,a_2,a_3} \mathfrak{a}_{a_1,a_2,a_3}^{(3)} \\
&= \mathfrak{a}^+ \cdot \left\langle \prod_{a_1,\ldots,a_{l+1}} \frac{\prod_{t \in 2\mathbf{Z}+1} h_{a_1,\ldots,a_t}^{(t)}}{\prod_{s \in 2\mathbf{Z}} h_{a_1,\ldots,a_s}^{(s)}} \right\rangle \cdot \underbrace{\prod_{a_1,\ldots,a_{l+1}} \mathfrak{a}_{a_1,\ldots,a_{l+1}}^{(l)}}_{=\mathfrak{a}^s}.
\end{aligned}$$

In this last expression, the indices are chosen such that $1 \leq t \leq l$ and $2 \leq s \leq l$. We also recall that all the quantities involved belong to the totally real subfield $\mathbf{Q}(\zeta + \zeta^{-1})$. By construction, the ideal $\mathfrak{a}^s$ is $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)$-smooth and we directly get $h \in \mathcal{O}_{\mathbf{K}}^+$ such that $\langle h \rangle = \mathfrak{a}^+ \cdot \mathfrak{a}^s$. The full outline of this descent phase is sketched in Figure 6. Remark that the number of terms, which is at most $O(n)^l = L_{|\Delta_{\mathbf{K}}|}(o(1))$, is negligible in the final complexity estimate.

6.3.3 Step 3: Case of $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)$-smooth ideals

At this point, we have reduced the search for a generator of a principal ideal of large norm to the search for a generator of a principal ideal $\mathfrak{a}^s$ which is $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)$-smooth. If we can find a generator of $\mathfrak{a}^s$ in time $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)$, from the previous steps we directly find a generator of $\mathfrak{a}^+$, and so a generator of $\mathfrak{a}$, that is the secret key. To tackle this final problem, we follow the same approach as in Chapter 5 relying on class group computation: we consider the previously introduced set $\mathcal{B}$ of prime ideals of norm below $B > 0$ where $B = Ł_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)$ and look for relations of the form

$$\langle v \rangle = \prod_i \mathfrak{p}_i^{e_i}, \qquad \text{for } v \in \mathcal{O}_{\mathbf{K}^+}.$$

[2] We can always run an additional step in the descent without changing the overall complexity.

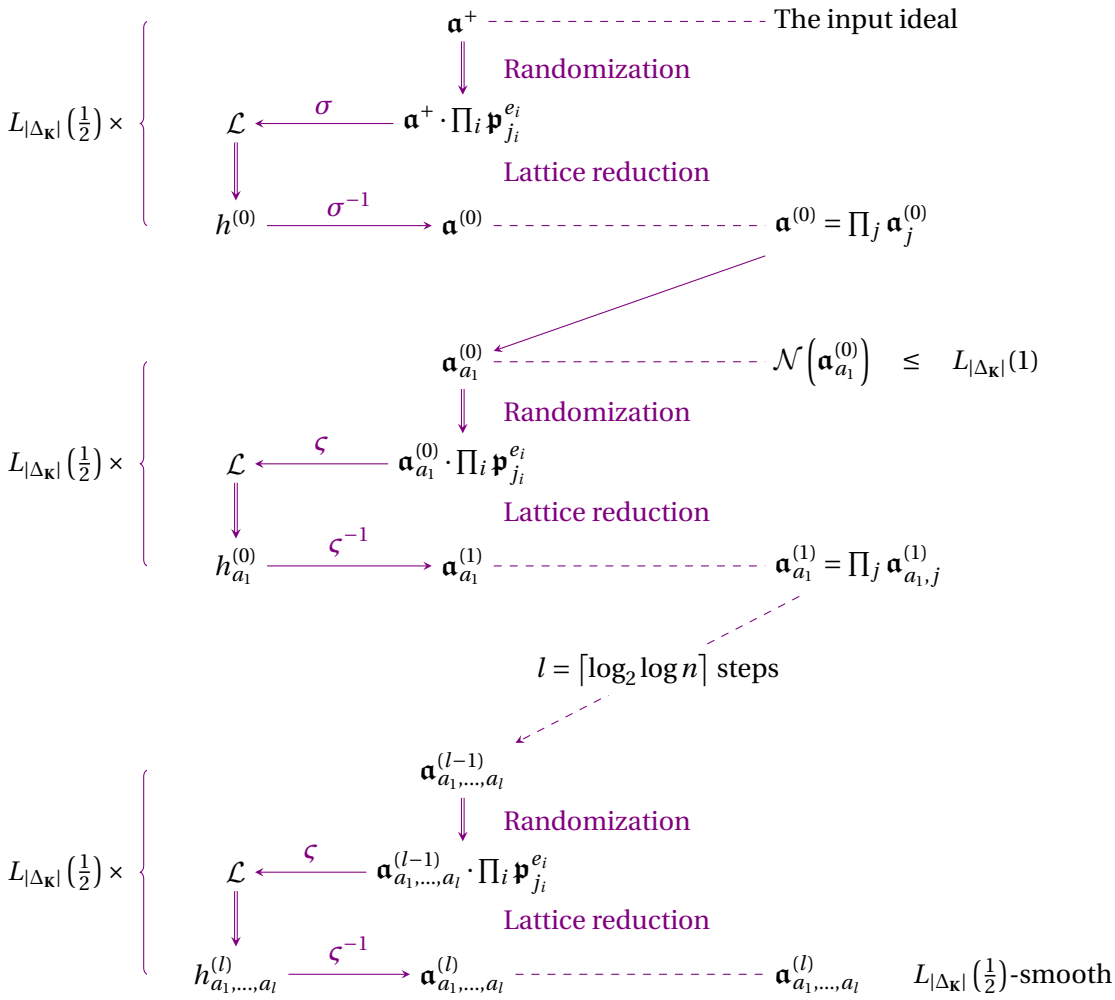

Figure 6: The descent algorithm.

The relation collection is performed in a similar way as in Chapter 5: due to the nice shape of the defining polynomial X^n+1, the algebraic integers whose representation as polynomials in ζ have small coefficients also have small norms. Let us fix an integer A such that $0 < A \leq L_{|\Delta_\mathbf{K}|}(0) = \log|\Delta_\mathbf{K}|$. Then for any integers $(v_0,\ldots,v_{\frac{n}{2}-1}) \in \{-A,\ldots,A\}^{\frac{n}{2}}$, we define the element $v = v_0 + \sum_{i\geq 1} v_i\left(\zeta^i + \zeta^{-i}\right)$. The norm of this element in $\mathbf{K}^+$ is upper bounded by $L_{|\Delta_\mathbf{K}|}(1)$. Indeed, it corresponds to the square root of its norm in $\mathbf{K}$, which is below $n^n \cdot A^n = L_{|\Delta_\mathbf{K}|}(1)$ by Lemma 1.4.4. Then under Heuristic 1.4.20, the element v generates an ideal $\langle v\rangle$ that is $L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}\right)$-smooth with probability $L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}\right)^{-1}$. This means that we need to draw on average $L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}\right)$ independent algebraic integers to find one relation.

To bound the runtime of the algorithm, we also need to assume Heuristic 4.2.2. It implies that there exists $K \ll L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}\right)$ such that collecting $K\cdot|\mathcal{B}|$ relations suffices to obtain a relation

matrix that has full rank. We conclude that $L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}\right)^2 = L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}\right)$ independently drawn algebraic integers suffice to generate a full-rank matrix. Of course, the set of algebraic integers arising from the previous construction is large enough to allow such repeated sampling, because its size is $L_{|\Delta_\mathbf{K}|}(1)$. We store the relations in a $K|\mathcal{B}| \times |\mathcal{B}|$ matrix M, and store the corresponding algebraic integers in a vector V.

$$\begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_{K|\mathcal{B}|} \end{pmatrix} \begin{matrix} \to \\ \to \\ \vdots \\ \to \end{matrix} \begin{pmatrix} M_{1,1} & \cdots & M_{1,|\mathcal{B}|} \\ M_{2,1} & \cdots & M_{2,|\mathcal{B}|} \\ \vdots & & \vdots \\ M_{K|\mathcal{B}|,1} & \cdots & M_{K|\mathcal{B}|,|\mathcal{B}|} \end{pmatrix} \quad \Longrightarrow \quad \forall i, \langle v_i \rangle = \prod_{j=1}^{|\mathcal{B}|} \mathfrak{p}_j^{M_{i,j}}.$$

The $L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}\right)$-smooth ideal $\mathfrak{a}^s$ splits over the set $\mathcal{B}$, so that there exists a vector Y in $\mathbf{Z}^{|\mathcal{B}|}$ containing the exponents of the factorization

$$\mathfrak{a}^s = \prod_i \mathfrak{p}_i^{Y_i}.$$

As the relations stored in M generate the lattice of all elements of this form, the vector Y necessarily belongs to this lattice. Hence solving the equation $MX = Y$ yields a vector $X \in \mathbf{Z}^{K|\mathcal{B}|}$. From this vector, we can recover a generator of the ideal since:

$$\prod_i \mathfrak{p}_i^{Y_i} = \left\langle v_1^{X_1} \cdots v_{K|\mathcal{B}|}^{X_{K|\mathcal{B}|}} \right\rangle. \tag{6.1}$$

By construction, $\mathcal{N}(\mathfrak{a}^s) \leq L_{|\Delta_\mathbf{K}|}\left(\frac{l+1}{2}\right)$ so that the coefficients of Y are bounded by $L_{|\Delta_\mathbf{K}|}(0)$. Since solving such a linear system with Dixon's p-adic method [Dix82] can be done in time $\mathrm{Poly}\left(d, \log\|M\|\right)$ where d is the dimension of the matrix and $\|M\| = \max|M_{i,j}|$ the maximum of its coefficients, we can find X in time $L_{|\Delta_\mathbf{K}|}\left(\frac{1}{2}\right)$.

6.3.4 Final Step: Reduction to a short generator

As mentioned in Section 6.1, this part of the algorithm is a result of Cramer, Ducas, Peikert, and Regev [CDPR16]. They state that finding a short generator given an arbitrary one can be done in polynomial time in any prime-power cyclotomic ring. For completeness, we present a brief overview of their reduction.

As a preliminary observation, note that for prime-power cyclotomic fields, a set of fundamental units is given for free, whereas their computation in arbitrary number fields is computationally hard. A second remark is that we know that there exists a small generator of the considered ideal. Then, instead of solving a general *closest vector problem* (CVP), we solve an instance of *bounded-distance decoding* problem (BDD). The key argument is based on a

precise study of the geometry of the Log-unit lattice of prime-power cyclotomic fields (see Section 1.5.2 for the required basic facts about units). Finally, the geometric properties of the units in those fields make it possible to solve the BDD problem in this lattice in polynomial time, instead of exponential time as for generic instances.

Theorem 6.3.3 ([CDPR16, Theorem 4.1])**.** *Let D be a distribution over $\mathbf{Q}(\zeta)$ with the property that for any tuple of vectors $v_1, \ldots, v_{\frac{n}{2}-1} \in \mathbf{R}^{\frac{n}{2}-1}$ of Euclidean norm 1 that are orthogonal to the all-1 vector $\mathbf{1}$, the probability that the inequality $\left|\left(\operatorname{Log} g, v_i\right)\right| < c\sqrt{2n} \cdot \log(2n)^{-\frac{3}{2}}$ holds for all i is at least some $\beta > 0$, where g is chosen from D and c is a universal constant. Then there is an efficient algorithm that, given $g' = g \cdot u$, where g is chosen from D and $u \in C$ is a cyclotomic unit, outputs an element of the form $\zeta^j \cdot g$ with probability at least β.*

Cyclotomic units. While giving the complete description of the units of a generic number field is a computationally hard problem, it is possible to describe a subgroup of finite index of the unit group in cyclotomic fields: the *cyclotomic units*. This subgroup contains all the units that are products of numbers[3] of the form $\zeta_m^i - 1$ for any $1 \leq i \leq m$. More precisely we have the following lemma.

Lemma 6.3.4 ([Was97, Lemma 8.1])**.** *Let m be a prime power, then the group C of cyclotomic units is generated by $\pm\zeta_m$ and $(b_i)_{1 \leq i \leq m}$, where*

$$b_i = \frac{\zeta_m^i - 1}{\zeta_m - 1}.$$

The index of the subgroup of cyclotomic units in the group of units is $h^+(m)$, the class number of the totally real subfield $\mathbf{Q}\left(\zeta_m + \zeta_m^{-1}\right)$ (see for instance [Was97]). In the case of power-of-two m, a well supported conjecture states that $h^+ = 1$.

Heuristic 6.3.5 (Weber's Class Number Problem)**.** *We assume that for power-of-two cyclotomic fields, the class number of its totally real subfield is 1.*

Thus, under Weber's heuristic, the cyclotomic units and the units coincide in the power-of-two cyclotomic fields.

The reader might argue that, in order to apply the previous theorem to the output of our algorithm, we should ensure that we produce a generator up to a *cyclotomic unit* and not up to an arbitrary unit. In the specific case of power-of-two cyclotomic fields, we can rely on Weber's heuristic (Heuristic 6.3.5) to make sure that this is indeed the case. In case $h^+ > 1$, two solutions are given in [CDPR16]. The first one is to directly compute the group

[3] One should notice that if m is a prime power, $\zeta_m^i - 1$ is not a unit, but b_i as in Lemma 6.3.4 is.

of units, which is hopefully determined by the kernel of the matrix M arising in the third stage (see Section 2.3). One can then enumerate the h^+ classes of the group of units modulo the subgroup of cyclotomic units. Another possibility is to generate a list of ideals, sampled according to the same distribution as the input ideal, with a known generator. Then, we apply the PIP algorithm to these ideals, and deduce the cosets of the group of units modulo the subgroup of cyclotomic units, which are likely to be output.

The complete key recovery, combining our PIP algorithm and the aforementioned reduction, is outlined in Algorithm 7.

Algorithm 7 Recovery of the secret key by PIP+[CDPR16].

Input: A $\mathbf{Z}$-basis of $\mathfrak{a}$.
Output: The secret key g up to a torsion unit.

1: Compute a generator g_0 of $\mathfrak{a}$ from the $\mathbf{Z}$-basis, using Gentry-Szydlo, descent and relation collection
2: Fix the basis B of the Log-unit lattice $\left(\operatorname{Log} b_i\right)_i$ with $b_i = \frac{\zeta_m^i - 1}{\zeta_m - 1}$
3: Set $t = \operatorname{Log} g_0 + \operatorname{Log} \mathcal{O}_{\mathbf{K}}$
4: **return** the rounding $B\lfloor (B^\vee)^t \cdot t \rceil$

6.4 Complexity analysis

The overall runtime of our attack is $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)$, that is about $2^{O\left(n^{\frac{1}{2}} \log n\right)}$ operations. We have already mentioned the complexity of most parts of our algorithm. However, we provide a brief summary in this section.

For the reduction algorithms, using BKZ and Cheon's trick, the block-size is always chosen as $\log L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)$ so that the complexity is $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)$. Our choice for the smoothness bound $B = Ł_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)$ ensures that time $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)$ suffices for the relation collection and linear system solution.

In addition, from the work of [GGH13], we get that the first part of the algorithm, corresponding to the reduction to the totally real subfield, is performed in polynomial time.

The last part, which corresponds to the generation of a small generator from an arbitrary one, runs in polynomial time with respect to the input (B, t) of the rounding algorithm (see Algorithm 7), thanks to the results of [CDPR16]. However, $t = \operatorname{Log} g_0 + \operatorname{Log} \mathcal{O}_{\mathbf{K}}$ is of subexponential size at this stage. Indeed, according to Equation (6.1),

$$\operatorname{Log} g_0 = X_1 \operatorname{Log} v_1 + \cdots + X_{K|\mathcal{B}|} \operatorname{Log} v_{K|\mathcal{B}|},$$

where each v_i is of polynomial size while, by Hadamard's bound, the integers X_i satisfy the relation $X_i \leq K|\mathcal{B}|^{\frac{K|\mathcal{B}|}{2}} \|M\|^{K|\mathcal{B}|-1} \max \|Y_j\|$. Therefore, the bit sizes of the X_i are $L_{|\Delta_{\mathbf{K}}|}\left(\frac{1}{2}\right)$, and

the fixed point approximations of $\operatorname{Log} v_i$ must be taken at precision $b = Ł_{|\Delta_K|}\left(\frac{1}{2}\right)$ to ensure the accuracy of the value of $\operatorname{Log} g_0$ (and therefore t). The rounding computation $B\lfloor (B^\vee)^t \cdot t \rceil$ has an asymptotic cost $L_{|\Delta_K|}\left(\frac{1}{2}\right)$ and returns $e_1, \ldots, e_r$ where the e_i have bit size $L_{|\Delta_K|}\left(\frac{1}{2}\right)$ and

$$g' = g_0 \cdot b_1^{e_1} \cdots b_r^{e_r} = \left(v_1^{X_1} \cdots v_{K|\mathcal{B}|}^{X_{K|\mathcal{B}|}}\right) \cdot \left(b_1^{e_1} \cdots b_r^{e_r}\right),$$

is a short generator of the input ideal. This product cannot be evaluated directly since the intermediate terms may have exponential size, but it may be performed modulo distinct prime ideals $\mathfrak{q}_1, \cdots, \mathfrak{q}_m$ such that $\mathcal{N}\left(\prod \mathfrak{q}_j\right) > \mathcal{N}(g')$ and then reconstructed by the Chinese Remainder Theorem. The complexity of this process is bounded by $L_{|\Delta_K|}\left(\frac{1}{2}\right)$.

As a consequence the overall complexity is dominated by the descent step, which runs in time $L_{|\Delta_K|}\left(\frac{1}{2}\right)$. Referring to the precise analysis in Section 5.5 to estimate the second constant, we are able to derive a precise complexity estimate. As mentioned in Section 4.1, prime-power cyclotomic fields — together with their totally real subfields — asymptotically belong to the class $\mathcal{D}_{1,1,1,0}$. Then the result stated at the very end of Section 5.5.2 implies that the complexity of this attack can be as low as

$$L_{|\Delta_K|}\left(\frac{1}{2}, \frac{\omega}{2\sqrt{2(\omega-1)}}\right).$$

Taking $\omega = \log_2 7$, we obtain a runtime for our attack of $L_{|\Delta_K|}\left(\frac{1}{2}, 0.738\right) = 2^{1.066 \cdot \sqrt{n} \log n}$.

Remark 6.4.1. This algorithm has a complexity in $L_{|\Delta_K|}\left(\frac{1}{2}\right)$ in the discriminant that represents the size of the number field involved. However, it is important to ascertain that the parameters of the keys have $n^{\frac{3}{2}}$ bits. Therefore we present an algorithm that is "sort of" $L\left(\frac{1}{3}\right)$ in the size of the inputs.

6.5 Implementation results

In addition to the theoretical improvement, our algorithm permits in practice to break concrete cryptosystems. Our discussion is based on the scheme presented by Smart and Vercauteren at PKC 2010. In [SV10, Section 7], security estimates are given for parameters $n = 2^m$ for $8 \leq m \leq 11$ since they were unable to generate keys for larger parameters. Our implementation allows us to find a secret key from the public key for $n = 2^8 = 256$ in less than a day: the code runs with `PARI/GP` [PARI], with an external call to `fplll` [fplll], and all the computations are performed on an Intel(R) Xeon(R) CPU E3-1275 v3 @ 3.50GHz with 32GB of memory. The large storage requirements are due to the Gentry-Szydlo algorithm.

We perform the key generation as in Algorithm 5. We then obtain a generator for the ideal as a polynomial in $\zeta = \zeta_{512}$, of degree 255 and coefficients absolutely bounded by $2^{\sqrt{256}}+1 = 65537$. That corresponds to ideals whose norms have about 4800 bits on average, which is below the

bound 6145 obtained from Lemma 1.4.4, but above the size given in [SV10] (4096). As for all timings in this section, we have derived a set of 10 keys, and the given time is the average one. Thus, deriving a secret key takes on average 30 seconds. We test 1381 algebraic integers resulting in ten that have a prime norm. Then the public key is derived from the secret key in about 96 seconds.

While, in theory, the first reduction to the totally real subfield seems to be of limited interest, it is clearly the main part of the practical results: indeed, it reduces in our example the size of the matrices involved from 256×256 to 128×128. As we know that lattice-reduction is getting worse while the dimension grows, this part is the key point of the algorithm. Our code essentially corresponds to the Gentry-Szydlo algorithm together with the trick explained in Section 6.3.1, in order to output the element u and a basis of the ideal $\mathfrak{a}^+$ generated by $g + \bar{g}$. This part of the algorithm has the largest runtime, about 20 hours, and requires 24GB of memory.

At this point, we put aside u and only consider the ideal $\mathfrak{a}^+$. Our goal is to recover one generator of this ideal, after which a multiplication by $\frac{1}{1+u}$ leads to a generator of the input ideal. The method we have presented is to reduce step by step the norms of the ideals involved by performing lattice reductions. However, we observe that for the cases we run, the first reduction suffices: the short vector we find corresponds to a generator. We make use of the BKZ algorithm implemented in `fplll` [fplll], with block-size 24 to begin with. It gives a correct generator with probability higher than 0.75 and runs in less than 10 minutes. If the output is not correct, we increase the block-size to 30. This always worked and required between 2 and 4 hours.

In addition to the good behavior of this reduction, the generator we found is already small, by construction. More precisely, it corresponds to $g + \bar{g}$, up to a factor that is a power of ζ. Hence, we recover $g \cdot \zeta^i$ thanks to u and the decoding algorithm analyzed in [CDPR16] turns out to be unnecessary for our application. The key recovery is already completed after these two first steps. Nevertheless, we implemented this part along with a method to find the actual private key (up to sign). Indeed, because all its coefficients are even except the constant one, it is easy to identify the power of ζ that appears as a factor during the computation.

Additional work. To illustrate the practical performances of our method, we look at one of the main other steps of the algorithm: namely the relation collection between generators of $Cl(\mathcal{O}_{\mathbf{K}^+})$. Thanks to the good behavior of BKZ, the relation collection is not necessary for the attack in $\mathbf{Q}(\zeta_{512})$, but it is an important part of the computation in higher dimension.

We fix our factor base as all the prime ideals in the totally-real field that lie above a prime number p that is below the bound $c\left(\log|\Delta_{\mathbf{K}}|\right)^2$, for a parameter $c \in \{0.1, 0.2, 0.3\}$. We list in

Table 4 the resulting bounds and number of prime ideals, along with the size of the factor base and the time required to construct the factor base in `MAGMA` [Magma]. The computations were performed on a laptop with Intel(R) Core(TM) i7-4710MQ CPU @ 2.50GHz and 8GB of RAM.

Naturally, this choice of bound would not be sufficient for the descent described in Section 6.3.2, because it is polynomial and not subexponential. However, it provides a relation matrix for the computation of the class group. Reaching a subexponential bound with the same process seems unlikely. It supports our suspicion that the good implementation results are a consequence of the small dimension obtained by the Gentry-Szydlo algorithm.

c	Bound	#primes	#Factor Base	Time (sec)
0.1	201516	149	18945	1240
0.2	403033	274	35073	2320
0.3	604549	385	49281	3320

Table 4: Construction of differently parametrized factor bases.

The relation collection is performed using algebraic integers of the form

$$\sum_{i=1}^{5} \zeta^{a_i} + \zeta^{-a_i} = \sum_{i=1}^{5} \zeta^{a_i} - \zeta^{256-a_i},$$

for a_i chosen at random in $\{1,\ldots,255\}$. This is inspired by the work of Miller [Mil14]. We use `C++` code with the NTL Library [NTL] to find a set of integers with different norms that suffice to generate the full lattice of relations (see Section 6.3.3). The size of these sets depends on the bound we have chosen and on the relations selected, so that the timings may vary. Our results are listed in Table 5. Once we know these integers, we use `MAGMA` to build the entire matrix of relations. In particular, we make use of the automorphisms on the field to derive 128 relations from each integer — this is the reason that we use integers of different norms. Eventually, the matrices we get are of full rank.

c	#relations	Time (hours)	
		relation collection	matrix construction
0.1	1500	8.6	1.7
0.2	3400	13.8	4.9
0.3	6300	23.9	10.7

Table 5: Relation collection for the different parameters.

We also ran our code for the algorithm described in [CDPR16] on inputs constructed as a secret key multiplied by a random non-zero vector of the Log-unit lattice (because in the full attack described previously, we only have the null vector). This took 150 seconds.

To conclude, for the parameter $n = 2^8$, the time of the key recovery is below 24 hours, and the main part of the computation comes from the reduction to the totally real subfield. Hence, one may wonder, again, if this step is mandatory, and, this time, the answer is an unqualified *"yes"*, because the surprisingly good practical behavior of the BKZ reduction is a consequence of the dimension of the lattices involved on the one hand — medium dimensions allow better practical output bounds than the theoretical worst case — and, on the other hand, the favorable properties of the geometry of the considered ideals induced by the abnormally small norms of their generators.

Bibliography

[AM93] Arthur O. L. Atkin and François Morain. Finding suitable curves for the elliptic curve method of factorization. *Mathematics of Computation*, 60:399–405, 1993.

[Bac90] Eric Bach. Explicit bounds for primality testing and related problems. *Mathematics of Computation*, 55:355–380, 1990.

[Bac95] Eric Bach. Improved approximations for Euler products. In *Number Theory, CMS Conference Proceedings*, volume 15, pages 13–28, 1995.

[BBB+12] Razvan Barbulescu, Joppe W. Bos, Cyril Bouvier, Thorsten Kleinjung, and Peter L. Montgomery. Finding ECM-friendly curves through a study of Galois properties. In *Proceedings of the 10th Algorithmic Number Theory Symposium ANTS 2012*, pages 63–86, 2012.

[BBHM02] Ingrid Biehl, Johannes A. Buchmann, Safuat Hamdy, and Andreas Meyer. A signature scheme based on the intractability of computing roots. *Designs, Codes and Cryptography*, 25(3):223–236, 2002.

[BBJ+08] Daniel J. Bernstein, Peter Birkner, Marc Joye, Tanja Lange, and Christiane Peters. Twisted Edwards curves. In *Progress in Cryptology - AFRICACRYPT 2008, Proceedings*, pages 389–405, 2008.

[BBL10] Daniel J. Bernstein, Peter Birkner, and Tanja Lange. Starfish on strike. In *Progress in Cryptology - LATINCRYPT 2010, Proceedings*, pages 61–80, 2010.

[BBLP13] Daniel J. Bernstein, Peter Birkner, Tanja Lange, and Christiane Peters. ECM using Edwards curves. *Mathematics of Computation*, 82(282):1139–1179, 2013.

[BBT94] Ingrid Biehl, Johannes A. Buchmann, and Christoph Thiel. Cryptographic protocols based on discrete logarithms in real-quadratic orders. In *Advances in Cryptology - CRYPTO 1994, Proceedings*, pages 56–60, 1994.

[BDF08] Karim Belabas, Francisco Diaz y Diaz, and Eduardo Friedman. Small generators of the ideal class group. *Mathematics of Computation*, 77(262):1185–1197, 2008.

[BDGL16] Anja Becker, Léo Ducas, Nicolas Gama, and Thijs Laarhoven. New directions in nearest neighbor searching with applications to lattice sieving. In *Proceedings of the 27th Annual ACM-SIAM Symposium on Discrete Algorithms SODA 2016*, pages 10–24, 2016.

[BEF+17] Jean-François Biasse, Thomas Espitau, Pierre-Alain Fouque, Alexandre Gélin, and Paul Kirchner. Computing generator in cyclotomic integer rings - A subfield algorithm for the Principal Ideal Problem in L(1/2) and application to the cryptanalysis of a FHE scheme. In *Advances in Cryptology - EUROCRYPT 2017, Proceedings*, pages 60–88, 2017.

[Bel04] Karim Belabas. Topics in computational algebraic number theory. *Journal de Théorie des Nombres de Bordeaux*, 16:19–63, 2004.

[BF14] Jean-François Biasse and Claus Fieker. Subexponential class group and unit group computation in large degree number fields. *LMS Journal of Computation and Mathematics*, 17:385–403, 2014.

[BGG+14] Johannes Blömer, Ricardo Gomes da Silva, Peter Günther, Juliane Krämer, and Jean-Pierre Seifert. A practical second-order fault attack against a real-world pairing implementation. In *2014 Workshop on Fault Diagnosis and Tolerance in Cryptography, FDTC 2014*, pages 123–136, 2014.

[BGN05] Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. Evaluating 2-DNF formulas on ciphertexts. In *Theory of Cryptography, Second Theory of Cryptography Conference, TCC 2005, Proceedings*, pages 325–341, 2005.

[BGV12] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (Leveled) fully homomorphic encryption without bootstrapping. In *Innovations in Theoretical Computer Science 2012*, pages 309–325, 2012.

[Bia14a] Jean-François Biasse. An L(1/3) algorithm for ideal class group and regulator computation in certain number fields. *Mathematics of Computation*, 83:2005–2031, 2014.

[Bia14b] Jean-François Biasse. Subexponential time relations in the class group of large degree number fields. *Advances in Mathematics of Communications*, 8(4):407–425, 2014.

[BJN+99] Johannes Buchmann, Michael J. Jacobson, Stefan Neis, Patrick Theobald, and Damian Weber. Sieving methods for class group computation. In *Algorithmic Algebra and Number Theory, Proceedings*, pages 3–10, 1999.

[BL10] Yuval Bistritz and Alexander Lifshitz. Bounds for resultants of univariate and bivariate polynomials. *Linear Algebra and its Applications*, 432:1995–2005, 2010.

[BLLN13] Joppe W. Bos, Kristin E. Lauter, Jake Loftus, and Michael Naehrig. Improved security for a ring-based fully homomorphic encryption scheme. In *Cryptography and Coding - 14th IMA International Conference, IMACC 2013, Proceedings*, pages 45–64, 2013.

[BS16] Jean-François Biasse and Fang Song. Efficient quantum algorithms for computing class groups and solving the principal ideal problem in arbitrary degree number fields. In *Proceedings of the 27th Annual ACM-SIAM Symposium on Discrete Algorithms SODA 2016*, pages 893–902, 2016.

[Buc87] Johannes Buchmann. The computation of the fundamental unit of totally complex quartic orders. *Mathematics of Computation*, 48(177):39–54, 1987.

[Buc90] Johannes Buchmann. A subexponential algorithm for the determination of class groups and regulators of algebraic number fields. *Séminaire de Théorie des Nombres, Paris 1988-1989*, pages 27–41, 1990.

[BW88] Johannes A. Buchmann and Hugh C. Williams. A key-exchange system based on imaginary quadratic fields. *Journal of Cryptology*, 1(2):107–118, 1988.

[BW89] Johannes Buchmann and Hugh C. Williams. On the complexity of the class number of an algebraic number field. *Mathematics of Computation*, 53(188):679–688, 1989.

[CD91] Henri Cohen and Francisco Diaz y Diaz. A polynomial reduction algorithm. *Journal de Théorie des Nombres de Bordeaux*, 3:351–360, 1991.

[CDPR16] Ronald Cramer, Léo Ducas, Chris Peikert, and Oded Regev. Recovering short generators of principal ideals in cyclotomic rings. In *Advances in Cryptology - EUROCRYPT 2016, Proceedings*, pages 559–585, 2016.

[CEP83] Earl R. Canfield, Paul Erdős, and Carl Pomerance. On a problem of Oppenheim concerning 'factorisatio numerorum'. *Journal of Number Theory*, 17:1–28, 1983.

[CGD] Gunter Malle. *Class Group Database*. http://www.mathematik.uni-kl.de/~numberfieldtables/.

[CGS14] Peter Campbell, Michael Groves, and Dan Shepherd. SOLILOQUY: A cautionary tale. ETSI 2nd Quantum-Safe Crypto Workshop, 2014. http://docbox.etsi.org/workshop/2014/201410_CRYPTO/S07_Systems_and_Attacks/S07_Groves.pdf.

[Che26] Nikolai G. Chebotarev. Die bestimmung der dichtigkeit einer menge von primzahlen, welche zu einer gegebenen substitutionsklasse gehören. *Mathematische Annalen*, 95:191–228, 1926.

[CL15] Jung Hee Cheon and Changmin Lee. Approximate algorithms on lattices with small determinant. Cryptology ePrint Archive, Report 2015/461, 2015. http://eprint.iacr.org/2015/461.

[CLG09] Denis Xavier Charles, Kristin E. Lauter, and Eyal Z. Goren. Cryptographic hash functions from expander graphs. *Journal of Cryptology*, 22(1):93–113, 2009.

[CLN16] Craig Costello, Patrick Longa, and Michael Naehrig. Efficient algorithms for supersingular isogeny Diffie-Hellman. In *Advances in Cryptology - CRYPTO 2016, Proceedings*, pages 572–601, 2016.

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

[DH76] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[Dic30] Karl Dickman. On the frequency of numbers containing prime factors of a certain relative magnitude. *Arkiv för Matematik, Astronomi och Fysik*, 22A(10):1–14, 1930.

[Dix82] John D. Dixon. Exact solution of linear quations using p-adic expansions. *Numerische Mathematik*, 40:137–141, 1982.

[Edw07] Harold M. Edwards. A normal form for elliptic curves. *Bulletin of the American Mathematical Society*, 44:393–422, 2007.

[EG07] Andreas Enge and Pierrick Gaudry. An L(1/3+ε) algorithm for the discrete logarithm problem for low degree curves. In *Advances in Cryptology - EUROCRYPT 2007, Proceedings*, pages 379–393, 2007.

[EGT11] Andreas Enge, Pierrick Gaudry, and Emmanuel Thomé. An L(1/3) discrete logarithm algorithm for low degree curves. *Journal of Cryptology*, 24:24–41, 2011.

[EJ17] Thomas Espitau and Antoine Joux. Practical reduction of quadratic forms with interval arithmetic. To appear, 2017.

[ElG84] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *Advances in Cryptology - CRYPTO 1984, Proceedings*, pages 10–18, 1984.

[ElG85] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, 1985.

[FJP14] Luca De Feo, David Jao, and Jérôme Plût. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. *Journal of Mathematical Cryptology*, 8(3):209–247, 2014.

[FO99] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In *Advances in Cryptology - CRYPTO 1999, Proceedings*, pages 537–554, 1999.

[Fon09] Felix Fontein. *The infrastructure of a global field and baby step-giant step algorithms*. PhD thesis, Universität Zürich, 2009. https://user.math.uzh.ch/fontein/diss-fontein.pdf.

[fplll] The FPLLL development team. *fplll, version 5.0*, 2016. https://github.com/fplll/fplll.

[FV12] Junfeng Fan and Frederik Vercauteren. Somewhat practical fully homomorphic encryption. Cryptology ePrint Archive, Report 2012/144, 2012. http://eprint.iacr.org/2012/144.

[Gal14] François Le Gall. Powers of tensors and fast matrix multiplication. In *International Symposium on Symbolic and Algebraic Computation ISSAC 2014*, pages 296–303, 2014.

[Gen09] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing STOC 2009*, pages 169–178, 2009.

[GGH13] Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. In *Advances in Cryptology - EUROCRYPT 2013, Proceedings*, pages 1–17, 2013.

[GJ16] Alexandre Gélin and Antoine Joux. Reducing number field defining polynomials: an application to class group computation. *LMS Journal of Computation and Mathematics*, 19:315–331, 2016.

[GJ17a] Alexandre Gélin and Antoine Joux. On the complexity of class group computations for large-degree number fields. *To appear*, 2017.

[GJ17b] Alexandre Gélin and Antoine Joux. Reducing the complexity for class group computations using small defining polynomials. *To appear*, 2017.

[GKL17] Alexandre Gélin, Thorsten Kleinjung, and Arjen K. Lenstra. Parametrizations for families of ECM-friendly curves. In *Proceedings of the ACM on International Symposium on Symbolic and Algebraic Computation, ISSAC 2017*, 2017. To appear.

[GLM09] Ying H. Gan, Cong Ling, and Wai H. Mow. Complex lattice reduction algorithm for low-complexity full-diversity MIMO detection. *IEEE Transactions on Signal Processing*, 57(7):2701–2710, 2009.

[GN08] Nicolas Gama and Phong Q. Nguyen. Finding short lattice vectors within Mordell's inequality. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing STOC 2008*, pages 207–216, 2008.

[GPS16] Steven D. Galbraith, Christophe Petit, and Javier Silva. Signature schemes based on supersingular isogeny problems. Cryptology ePrint Archive, Report 2016/1154, 2016. http://eprint.iacr.org/2016/1154.

[GPST16] Steven D. Galbraith, Christophe Petit, Barak Shani, and Yan Bo Ti. On the security of supersingular isogeny cryptosystems. In *Advances in Cryptology - ASIACRYPT 2016, Proceedings*, pages 63–91, 2016.

[GS02] Craig Gentry and Michael Szydlo. Cryptanalysis of the revised NTRU signature scheme. In *Advances in Cryptology - EUROCRYPT 2002, Proceedings*, pages 299–320, 2002.

[GW17] Alexandre Gélin and Benjamin Wesolowski. Loop-abort faults on supersingular isogeny cryptosystems. In *Post-Quantum Cryptography - 8th International Workshop, PQCrypto 2017, Proceedings*, 2017. To appear.

[HM89] James L. Hafner and Kevin S. McCurley. A rigorous subexponential algorithm for computation of class groups. *Journal of American Mathematical Society*, 2:839–850, 1989.

[HM00] Safuat Hamdy and Bodo Möller. Security of cryptosystems based on class groups of imaginary quadratic orders. In *Advances in Cryptology - ASIACRYPT 2000, Proceedings*, pages 234–247, 2000.

[HPS98] Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. NTRU: A ring-based public key cryptosystem. In *Proceedings of the 3rd Algorithmic Number Theory Symposium ANTS 1998*, pages 267–288, 1998.

[HPS11] Guillaume Hanrot, Xavier Pujol, and Damien Stehlé. Analyzing blockwise lattice algorithms using dynamical systems. In *Advances in Cryptology - CRYPTO 2011, Proceedings*, pages 447–464, 2011.

[HS07] Guillaume Hanrot and Damien Stehlé. Improved analysis of Kannan's shortest lattice vector algorithm. In *Advances in Cryptology - CRYPTO 2007, Proceedings*, pages 170–186, 2007.

[HS08] Guillaume Hanrot and Damien Stehlé. Worst-case Hermite-Korkine-Zolotarev reduced lattice bases. arXiv:0801.3331, 2008. https://arxiv.org/pdf/0801.3331.pdf.

[HT93] Adolf Hildebrand and Gerald Tenenbaum. Integers without large prime factors. *Journal de Théorie des Nombres de Bordeaux*, 5(2):411–484, 1993.

[HWCD08] Huseyin Hisil, Kenneth Koon-Ho Wong, Gary Carter, and Ed Dawson. Twisted Edwards curves revisited. In *Advances in Cryptology - ASIACRYPT 2008, Proceedings*, pages 326–343, 2008.

[JF11] David Jao and Luca De Feo. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In *Post-Quantum Cryptography - 4th International Workshop PQCrypto 2011, Proceedings*, pages 19–34, 2011.

[JS14] David Jao and Vladimir Soukharev. Isogeny-based quantum-resistant undeniable signatures. In *Post-Quantum Cryptography - 6th International Workshop PQCrypto 2014, Proceedings*, pages 160–179, 2014.

[KAKJ17] Brian Koziel, Reza Azarderakhsh, Mehran Mozaffari Kermani, and David Jao. Post-quantum cryptography on FPGA based on isogenies on elliptic curves. *IEEE Transactions on Circuits and Systems*, 64(1):86–99, 2017.

[Kan83] Ravi Kannan. Improved algorithms for integer programming and related lattice problems. In *Proceedings of the 15th Annual ACM Symposium on Theory of Computing STOC 1983*, pages 99–108, 1983.

[KGV16] Alhassan Khedr, P. Glenn Gulak, and Vinod Vaikuntanathan. SHIELD: scalable homomorphic implementation of encrypted data-classifiers. *IEEE Transactions on Computers*, 65(9):2848–2858, 2016.

[KJA+16] Brian Koziel, Amir Jalali, Reza Azarderakhsh, David Jao, and Mehran Mozaffari Kermani. NEON-SIDH: efficient implementation of Supersingular Isogeny Diffie-Hellman key exchange protocol on ARM. In *Cryptology and Network Security - 15th International Conference, CANS 2016, Proceedings*, pages 88–103, 2016.

[KLM+15] Daniel Kirkwood, Bradley C. Lackey, John McVey, Mark Motley, Jerome A. Solinas, and David Tuller. Failure is not an option: Standardization issues for post-quantum key agreement on the security of supersingular isogeny cryptosystems. Workshop on Cybersecurity in a Post-Quantum World, 2015. http://csrc.nist.gov/groups/ST/post-quantum-2015/presentations/session7-motley-mark.pdf.

[Kob87] Neal Koblitz. Elliptic curve cryptosystems. *Mathematics of Computation*, 48(177):203–209, 1987.

[KP76] Donald E. Knuth and Luis Trabb Pardo. Analysis of a simple factorization algorithm. *Theoretical Computer Science*, 3(3):321–348, 1976.

[KPG99] Aviad Kipnis, Jacques Patarin, and Louis Goubin. Unbalanced oil and vinegar signature schemes. In *Advances in Cryptology - EUROCRYPT 1999, Proceedings*, pages 206–222, 1999.

[Lan03] Edmund Landau. Neuer Beweis des Primzahlsatzes und Beweis des Primidealsatzes. *Mathematische Annalen*, 56:645–670, 1903.

[Leh33] Derrick H. Lehmer. Factorization of certain cyclotomic functions. *Annals of Mathematics*, 34:461–479, 1933.

[Leh88] Emma Lehmer. Connection between Gaussian periods and cyclic units. *Mathematics of Computation*, 50(182):535–541, 1988.

[Len82] Hendrik W. Lenstra Jr. On the calculation of regulators and class numbers of quadratic fields. In *Journées Arithmétiques 1980*, pages 123–150, 1982.

[Len87] Hendrik W. Lenstra Jr. Factoring integers with elliptic curves. *Annals of Mathematics*, 126:649–673, 1987.

[LL93] Arjen K. Lenstra and Hendrik W. Lenstra Jr. *The development of the number field sieve*, volume 1554 of *Lecture Notes in Mathematics*. Springer-Verlag, 1993.

[LLL82] Hendrik W. Lenstra Jr., Arjen K. Lenstra, and László Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261:515–534, 1982.

[LLMP90] Arjen K. Lenstra, Hendrik W. Lenstra Jr., Mark S. Manasse, and John M. Pollard. The number field sieve. In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing STOC 1990*, pages 564–572, 1990.

[LO77] Jeffrey C. Lagarias and Andrew M. Odlyzko. Effective versions of the Chebotarev theorem. *Algebraic Number Fields*, pages 409–464, 1977.

[LOM79] Jeffrey C. Lagarias, Andrew M. Odlyzko, and Hugh L. Montgomery. A bound for the least prime ideal in the Chebotarev density theorem. *Inventiones Mathematicae*, 54(3):271–296, 1979.

[LPR13] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. *Journal of the ACM*, 60(6), 2013.

[LS14] Hendrik W. Lenstra and Alice Silverberg. Revisiting the Gentry-Szydlo algorithm. In *Advances in Cryptology - CRYPTO 2014, Proceedings*, pages 280–296, 2014.

[LSS14] Adeline Langlois, Damien Stehlé, and Ron Steinfeld. GGHLite: More efficient multilinear maps from ideal lattices. In *Advances in Cryptology - EUROCRYPT 2014, Proceedings*, pages 239–256, 2014.

[Magma] Computational Algebra Group, University of Sydney. *MAGMA, version 2.21.6*, 2016. http://magma.maths.usyd.edu.au/magma/.

[Mah60] Kurt Mahler. An application of Jensen's formula to polynomials. *Mathematika*, 7:98–100, 1960.

[Mah64] Kurt Mahler. An inequality for the discriminant of a polynomial. *Michigan Mathematical Journal*, 11:257–262, 1964.

[McC89] Kevin S. McCurley. Cryptographic key distribution and computation in class groups. *Number Theory and Apllications*, 265:459–479, 1989.

[McE78] Robert J. McEliece. A public-key cryptosystem based on algebraic coding theory. *DSN Progress Report*, 44:114–116, 1978.

[MG94] Maurice Mignotte and Philippe H. Glesser. Landau's inequality via Hadamard's. *Journal of Symbolic Computation*, 18(4):379–383, 1994.

[MI88] Tsutomu Matsumoto and Hideki Imai. Public quadratic polynominal-tuples for efficient signature-verification and message-encryption. In *Advances in Cryptology - EUROCRYPT 1988, Proceedings*, pages 419–453, 1988.

[Mil85] Victor S. Miller. Use of elliptic curves in cryptography. In *Advances in Cryptology - CRYPTO 1985, Proceedings*, pages 417–426, 1985.

[Mil14] John C. Miller. Class numbers of totally real fields and applications to the Weber class number problem. *Acta Arithmetica*, 164:381–397, 2014.

[MNP01] Andreas Meyer, Stefan Neis, and Thomas Pfahler. First implementation of cryptographic protocols based on algebraic number fields. In *Information Security and Privacy, 6th Australasian Conference, ACISP 2001, Proceedings*, pages 84–103, 2001.

[Moi30] Abraham De Moivre. Miscellanea analytica de seriebus et quadraturis. London, 1730.

[Mon87] Peter L. Montgomery. Speeding the Pollard and elliptic curve methods of factorization. *Mathematics of Computation*, 48:243–264, 1987.

[MW16] Daniele Micciancio and Michael Walter. Practical, predictable lattice basis reduction. In *Advances in Cryptology - EUROCRYPT 2016, Proceedings*, pages 820–849, 2016.

[NTL] Victor Shoup. *NTL: A Library for doing Number Theory, version 9.11.0*, 2016. http://http://www.shoup.net/ntl/.

[Oes79] Joseph Oesterlé. Versions effectives du théorème de Chebotarev sous l'hypothèse de Riemann généralisée. *Astérique*, 61:165–167, 1979.

[Pai99] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Advances in Cryptology - EUROCRYPT 1999, Proceedings*, pages 223–238, 1999.

[PARI] The PARI Group, Bordeaux. *PARI/GP, version 2.7.6*, 2016. http://pari.math.u-bordeaux.fr/.

[Pei09] Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing STOC 2009*, pages 333–342, 2009.

[Poh93] Michael E. Pohst. *Computational algebraic number theory*, volume 21 of *DMV Lecture Notes*. Birhaüser, Basel, 1993.

[PT00] Sachar Paulus and Tsuyoshi Takagi. A new public-key cryptosystem over a quadratic order with quadratic decryption time. *Journal of Cryptology*, 13(2):263–272, 2000.

[PV06] Dan Page and Frederik Vercauteren. A fault attack on pairing-based cryptography. *IEEE Transactions on Computers*, 55(9):1075–1080, 2006.

[PZ89] Michael Pohst and Hans Zassenhaus. *Algorithmic Algebraic Number Theory*, volume 30 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, Cambridge, 1989.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing STOC 2005*, pages 84–93, 2005.

[RSA78] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commununications of the ACM*, 21(2):120–126, 1978.

[SBW94] Renate Scheidler, Johannes A. Buchmann, and Hugh C. Williams. A key-exchange protocol using real quadratic fields. *Journal of Cryptology*, 7(3):171–199, 1994.

[Sch82] René Schoof. A hierarchy of polynomial time lattice basis reduction algorithms. *Mathematisch Centrum Computational Methods in Number Theory*, pages 235–286, 1982.

[Sch91] Claus-Peter Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology*, 4(3):161–174, 1991.

[Sch15] John Schanck. LOGCVP, Pari implementation of CVP in $\log \mathbb{Z}[\zeta_{2^n}]^*$, 2015. https://github.com/jschanck-si/logcvp.

[Sco04] Eira Scourfield. On ideals free of large prime factors. *Journal de Théorie des Nombres de Bordeaux*, 16(3):733–772, 2004.

[Sey87] Martin Seysen. A probabilistic factorization algorithm with quadratic forms of negative discriminant. *Mathematics of Computation*, 84:757–780, 1987.

[Sha69] Daniel Shanks. Class number, a theory of factorization, and genera. In *Proceedings of Symposia in Pure Mathematics*, volume 20, pages 415–440, 1969.

[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, 1997.

[SIDH] Microsoft Security and Cryptography. *SIDH Library*, 2016. https://www.microsoft.com/en-us/research/project/sidh-library/.

[SL96] Arne Storjohann and George Labahn. Asymptotically fast computation of Hermite normal forms of integer matrices. In *Proceedings of the International Symposium on Symbolic and Algebraic Computation ISSAC 1996*, pages 259–266, 1996.

[Ste75] Hans-Joachim Stender. Eine Formel für Grundeinheiten in reinen algebraischen Zahlkörpern dritten, vierten und sechsten Grades. *Journal of Number Theory*, 7(2):235–250, 1975.

[Sti30] James Stirling. Methodus differentialis. London, 1730.

[Sto05] Arne Storjohann. The shifted number system for fast linear algebra on integer matrices. *Journal of Complexity*, 21(4):609–650, 2005.

[Str69] Volker Strassen. Gaussian elimination is not optimal. *Numerische Mathematik*, 13:354–356, 1969.

[STW12] Xi Sun, Haibo Tian, and Yumin Wang. Toward quantum-resistant strong designated verifier signature from isogenies. In *4th International Conference on Intelligent Networking and Collaborative Systems INCoS 2012*, pages 292–296, 2012.

[Suy85] Hiromi Suyama. Informal preliminary report. (cited in [Mon87]), 1985.

[SV10] Nigel P. Smart and Frederik Vercauteren. Fully homomorphic encryption with relatively small key and ciphertext sizes. In *Public Key Cryptography - PKC 2010, Proceedings*, pages 420–443, 2010.

[Thi95] Christoph Thiel. *On the complexity of some problems in algorithmic algebraic number theory*. PhD thesis, Universität des Saarlandes,

1995. https://www.cdc.informatik.tu-darmstadt.de/reports/reports/Christoph_Thiel.diss.pdf.

[Was97] Lawrence C. Washington. *Introduction to Cyclotomic Fields*, volume 83 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 2nd edition, 1997.

[Wil95] Andrew Wiles. Modular elliptic curves and Fermat's Last Theorem. *Annals of Mathematics*, 142(3):443–551, 1995.

Abstract

A common way to construct public-key cryptosystems is to use the discrete exponentiation in a finite group where the inverse problem, the discrete logarithm problem, is considered difficult. Several groups are commonly used: multiplicative group of finite fields or Jacobians of algebraic curves defined over finite fields. The class group of a number field is another candidate which was sometimes proposed. To study its viability, it is important to identify the difficulty for this kind of group to determine its cardinality, its structure and the possible hardness of the discrete logarithm problem. Irrespective of cryptologic applications, methods to determine the properties of these groups is of independent mathematical interest.

In this thesis, we focus on class group computations in number fields. We start by describing an algorithm for reducing the size of a defining polynomial of a number field. There exist infinitely many polynomials that define a specific number field, with arbitrarily large coefficients, but our algorithm constructs the one that has the absolutely smallest coefficients. The advantage of knowing such a "small" defining polynomial is that it makes calculations in the number field easier because smaller values are involved. In addition, thanks to such a small polynomial, one can use specific algorithms that are more efficient than the general ones for class group computations.

The generic algorithm to determine the structure of a class group is based on ideal reduction, where ideals are viewed as lattices. We describe and simplify the algorithm presented by Biasse and Fieker in 2014 at ANTS and provide a more thorough complexity analysis for it. We also examine carefully the case of number fields defined by a polynomial with small coefficients. We describe an algorithm similar to the Number Field Sieve, which, depending on the field parameters, may reach the hope for complexity $L\left(\frac{1}{3}\right)$.

Finally, our results can be adapted to solve an associated problem: the Principal Ideal Problem. Given any basis of a principal ideal (generated by a unique element), we are able to find such a generator. As this problem, known to be hard, is the key-point in several homomorphic cryptosystems, the slight modifications of our algorithms provide efficient attacks against these cryptographic schemes.

Résumé

L'une des voies privilégiées pour la construction de cryptosystèmes à clé publique est l'utilisation dans des groupes finis de l'exponentiation discrète, dont le problème inverse, celui du logarithme discret, est réputé difficile. Plusieurs groupes sont classiquement utilisés : le groupe multiplicatif d'un corps fini ou la Jacobienne d'une courbe algébrique définie sur un corps fini. Le groupe de classes d'un corps de nombres est un autre candidat qui a parfois été proposé. Pour étudier sa viabilité, il est important de bien cerner la difficulté de déterminer la cardinalité et la structure du groupe, ainsi que la difficulté éventuelle du problème du logarithme discret. Indépendamment de ces applications cryptographiques, les méthodes utilisées pour résoudre ces problèmes ont aussi un intérêt mathématique.

Dans cette thèse, nous nous intéressons au calcul du groupe de classes d'un corps de nombres. Nous débutons par décrire un algorithme de réduction du polynôme de définition d'un corps de nombres. Il existe une infinité de polynômes qui définissent un corps de nombres fixé, avec des coefficients arbitrairement gros. Notre algorithme calcule celui qui a les plus petits coefficients. L'avantage de connaître un petit polynôme de définition est qu'il simplifie les calculs entre éléments de ce corps de nombres, en impliquant des quantités plus petites. En outre, la connaissance d'un tel polynôme permet l'utilisation d'algorithmes plus efficaces que dans le cas général pour calculer le groupe de classes.

L'algorithme général pour calculer la structure du groupe de classes repose sur la réduction d'idéaux, vus comme des réseaux. Nous décrivons et simplifions l'algorithme présenté par Biasse et Fieker en 2014 à ANTS et approfondissons l'analyse de complexité. Nous nous sommes aussi intéressés au cas des corps de nombres définis par un polynôme à petits coefficients. Nous décrivons un algorithme similaire au crible par corps de nombres (NFS) dont la complexité en fonction des paramètres du corps de nombres peut atteindre $L(\frac{1}{3})$.

Enfin, nos algorithmes peuvent être adaptés pour résoudre un problème lié : le Problème de l'Idéal Principal. Étant donné n'importe quelle base d'un idéal principal (généré par un seul élément), nous sommes capables de retrouver ce générateur. Cette application de nos algorithmes fournit une attaque efficace contre certains schémas de chiffrement homomorphe basés sur ce problème.