Sieving Methods for Class Group Computation

Johannes Buchmann, Michael J. Jacobson, Jr., Stefan Neis, Patrick Theobald, and Damian Weber

Institut für Theoretische Informatik
Technische Universität Darmstadt

1 Introduction

Computing the class group and regulator of an algebraic number field K are two major tasks of algorithmic algebraic number theory. The asymptotically fastest method known has conjectured sub-exponential running time and was proposed in [Buc89]. In this paper we show how sieving methods developed for factoring algorithms can be used to speed up this algorithm in practice. We present numerical experiments which demonstrate the efficiency of our new strategy. For example, we are able to compute the class group of an imaginary quadratic field with a discriminant of 55 digits 20 times as fast as S. Düllmann in an earlier record–setting implementation ([BD91a]) which did not use sieving techniques. We also present class numbers of large cubic fields.

2 The Algorithm

We will consider the problem of computing the class group $Cl(K)$ of an algebraic number field K given by an irreducible monic polynomial of degree $n = r + 2s$, where r is the number of real embeddings and s is the number of complex embeddings of K into the field $\mathbb{C}$ of complex numbers. We denote the maximal order of K by $\mathcal{O}_K$. The norm of an algebraic number α will be denoted by $\mathcal{N}(\alpha)$, and the norm of an ideal $\mathfrak{a}$ will be denoted by $\mathcal{N}(\mathfrak{a})$. The class number of K will be denoted by h and the regulator by R.

We briefly review the algorithm presented in [Buc89]. Let FB be a set of prime ideals over K and $k = |FB|$. For $\boldsymbol{e} = (e_1, \dots, e_k) \in \mathbb{Z}^k$, we write

$$FB^{\boldsymbol{e}} = \prod_{i=1}^{k} \mathfrak{p}_i^{e_i}.$$

By A_{FB} we denote all algebraic numbers α in K which, considered as principal ideals, can be represented as a power product of the ideals of the *factor base* FB, i.e.,

$$A_{FB} = \left\{ \alpha \in K \mid \alpha \cdot \mathcal{O}_K = FB^{\boldsymbol{e}}, e \in \mathbb{Z}^k \right\}.$$

Consider the maps

$$\begin{aligned} \Phi' : A_{FB} &\longrightarrow \mathbb{Z}^k \\ \alpha &\longmapsto \boldsymbol{e} \end{aligned}$$

and

$$\begin{aligned}\Phi : A_{FB} &\longrightarrow \mathbb{Z}^k \times \mathbb{R}^{r+s-1}\\ \alpha &\longmapsto (\boldsymbol{e}, \log|\sigma_1(\alpha)|, \ldots, \log|\sigma_{r+s-1}(\alpha)|),\end{aligned}$$

where $\alpha \cdot \mathcal{O}_K = FB^{\boldsymbol{e}}$ and the σ_i are the embeddings of K into $\mathbb{C}$.

By [Buc89, Theorem 2.1] we know:

Theorem 1. *Suppose that the ideal classes of the elements of FB generate the class group of K. Then $\Phi(A_{FB})$ is a $(k+r+s-1)$–dimensional lattice with determinant hR. Also, $\Phi'(A_{FB})$ is a k–dimensional lattice with determinant h and we have $\mathbb{Z}^k/\Phi'(A_{FB}) \cong Cl(K)$.*

Based on this theorem, we can compute the class number and regulator by finding *relations*, i.e., algebraic numbers α together with their decompositions over the factor base. Having found enough relations to generate a sub-lattice of $\Phi(A_{FB})$ of full dimension, it can be checked whether these relations generate the full lattice by applying the analytic class number formula

$$\frac{2^r(2\pi)^s}{w\sqrt{|\Delta_K|}}hR = \prod_{p\text{prime}} \frac{1-\frac{1}{p}}{\prod_{\mathfrak{p}|p} 1-\frac{1}{\mathcal{N}(\mathfrak{p})}},$$

where Δ_K denotes the discriminant of the field K and w denotes the number of roots of unity in K. This formula enables us to compute a number h^* with

$$h^* < hR < 2h^*.$$

If the determinant of the sublattice of $\Phi(A_{FB})$ is in this interval then the sublattice is the full lattice. Otherwise, additional relations can be generated until we find the full lattice.

3 Generating Relations

Several methods for finding relations among the factor base elements have been suggested, for example, in [BD91a], [PZ89], and [CDyDO93]. We show how to use sieving techniques known from factoring algorithms to compute relations very efficiently.

3.1 Quadratic Fields

For quadratic fields, we can apply a modification of the multiple polynomial quadratic sieve (for details, see [Jac97]). Suppose that we find $\alpha \in K$, $\boldsymbol{v}, \boldsymbol{e} \in \mathbb{Z}^k$ such that $(\alpha)/FB^{\boldsymbol{v}} = FB^{\boldsymbol{e}}$. If K is imaginary quadratic then $(\boldsymbol{e}+\boldsymbol{v}) \in \Phi'(A_{FB})$. If K is real quadratic then $(\boldsymbol{e}+\boldsymbol{v}, \log|\sigma_1(\alpha)|) \in \Phi(A_{FB})$. To find such numbers α, we proceed as follows:

- Pick a random exponent $\boldsymbol{v} \in \{-1,0,1\}^k$ and compute the ideal $\mathfrak{a} = FB^{\boldsymbol{v}}$. Let $\mathfrak{a} = a\mathbb{Z} + \frac{b+\sqrt{\Delta}}{2}\mathbb{Z}$. Then, all elements in $\mathfrak{a}$ have norm $af(x,y)$, $x,y \in \mathbb{Z}$, where $f = aX^2 + bXY + cY^2 \in \mathbb{Z}[X,Y]$ and $c = (b^2 - \Delta)/(4a)$.
- Find integers x such that $f(x,1) = \prod_{i=1}^{k} \mathcal{N}(\mathfrak{p}_i)^{\overline{e}_i}$.
- For each such x, set $\alpha = ax + \frac{b+\sqrt{\Delta}}{2}$. Then $\mathcal{N}(\alpha) = af(x,1)$ is valid.
- Factor $(\alpha)/\mathfrak{a} =: \mathfrak{b} = FB^{\boldsymbol{e}}$, using the fact that $\mathcal{N}(\mathfrak{b}) = n$ and thus, $e_i = \pm\overline{e}_i$. We obtain a relation for each such integer x.

Note that this strategy has very much in common with the relation generation part of the multiple polynomial quadratic sieve factoring algorithm [Sil87]. In both cases, the heart of the method is finding smooth values of a quadratic polynomial. As a result, it is not difficult to apply many of the tricks used in implementing the factoring algorithm in our case. For example, we can easily select the leading coefficient of our polynomials to be a square-free product of small primes and be about as large as $\frac{\sqrt{|\Delta|}}{M}$, where we will sieve the polynomial over $x \in (-M, M)$. This has the effect of minimizing the values of the polynomial over this interval and forces our relation matrix to be sparse. See [Jac97] for more details.

3.2 The General Case

In the case of number fields of arbitrary degree, we may apply the sieving procedure of the number field sieve instead of the quadratic sieve.

As was the case for the quadratic sieve, the number field sieve was originally invented for factoring integers (see [BLP93]). A variation of this algorithm has been implemented by D. Weber to compute discrete logarithms in finite prime fields ([Web97]). By abandoning the need to simultaneously find smooth numbers in two number rings, we can use the number field sieve to find algebraic integers of smooth norm in the number field for which we want to compute the class group, i.e., algebraic integers with norm that factors over a given factor base. Given a number field K of degree n with generating polynomial f, we can use this tool to find smooth principal ideals and their decomposition

$$(x + y\rho)\mathcal{O} = (FB')^{\boldsymbol{e}'}$$

in the order $\mathcal{O} = \mathbb{Z}[X]/(f)$, where FB' is a factor base consisting of prime ideals of $\mathcal{O}$ and ρ is a zero of f. These factorizations can be lifted to factorizations in $\mathcal{O}_K$ by computing $\mathrm{ord}_{\mathfrak{p}}(\mathfrak{a})$ for the index divisors $\mathfrak{p} \mid [\mathcal{O}_K : \mathcal{O}]$. In this way, we obtain factorizations

$$(x + y\rho)\mathcal{O}_K = (FB)^{\boldsymbol{e}},$$

and vectors

$$(\boldsymbol{e}, \log|\sigma_1(\alpha)|, \ldots, \log|\sigma_{r+s-1}(\alpha)|) \in \Phi(A_{FB}).$$

However, this method still seems to be impractical for large discriminants, since the lattice generated by the relations usually does not have full rank. Therefore, we also try to carry over the idea described for quadratic fields. In general, we will proceed as follows:

- Select a random $\boldsymbol{v} \in \{-10, \ldots, 0, \ldots, 10\}^k$ and compute $\mathfrak{a} = FB^{\boldsymbol{v}}$. Often $\boldsymbol{v}$ will contain a certain prescribed non–zero entry at a position corresponding to a linearly dependent row of the relation matrix. In most cases, this will increase the rank of the relation matrix.
- Find $\alpha, \beta \in \mathfrak{a}$ of small height and thus, not too large norm.
- Compute $f = \mathcal{N}(\alpha X + \beta Y)/\mathcal{N}(\mathfrak{a})$. f is a homogeneous polynomial in $\mathbb{Z}[X, Y]$ of degree n.
- Find integers x and y such that $f(x, y) = \prod_{i=1}^{k} p_i^{\bar{e}_i}$, where all the prime factors of $f(x, y)$ are rational primes contained in the prime ideals in the factor base.
- For each such pair (x, y) set $\gamma = \alpha x + \beta y$. Then $\mathcal{N}(\gamma) = \mathcal{N}(\mathfrak{a}) f(x, y)$ is valid.
- Check if $(\gamma)/\mathfrak{a} =: \mathfrak{b}$ factors over the original factor base. If yes, we obtain a relation for this pair (x, y).

In practice, one of the main problems of this method is to obtain polynomials f that generate many smooth values. This problem consists of two parts. First, we want a good generating polynomial for the number field, i.e., typically one with small coefficients and with an equation order of small index. Second, we want to choose α and β in such a way that $\mathcal{N}(\alpha X + \beta Y)/\mathcal{N}(\mathfrak{a})$ is well-suited for sieving. It is well–known that finding polynomials suitable for the number field sieve is a hard problem. It is even unknown how to check whether a given polynomial will turn out to be good without actually trying it. In our case, we have to use many polynomials and therefore cannot afford to test every polynomial we compute, so we developed some heuristics to rule out the bad polynomials beforehand. Moreover, by reducing the ideal $\mathfrak{a}$ and doing the remaining steps in the reduced ideal, we can greatly improve the quality of the generated polynomials. Those techniques will be explained in more detail in [BNW].

4 HNF Computation

After finding enough relations, we have to compute the Hermite normal form of the lattice generated by the relations. Since this requires linear algebra over $\mathbb{Z}$, this problem is considerably harder than the linear algebra step involved in factoring or discrete logarithm computation. However, since the matrices involved are sparse and contain small entries, it is possible to apply special techniques to do this computation as follows.

In the first stage, we apply an algorithm for computing the Hermite normal form using unimodular operations over $\mathbb{Z}$. This algorithm is basically described in [Coh95] and computes the HNF successively row by row. To conserve sparseness and avoid entry explosion we use several heuristics in the pivot selection and entry elimination step. These heuristics are based on results of [HM91], [HHR93], [HM94a], [HM94b] and a lot of practical experience. The heuristics used will be adapted, depending on the number of entries in the remaining part of the matrix and the size of these entries.

If the entries become too large or the matrix becomes "too dense" (for example, if more than two thirds of the remaining entries are non–zero), we switch to stage two. We compute a multiple of the lattice determinant of the relation matrix. In this step, we try to obtain as small a multiple as possible to simplify the following computation. Finally we apply the modular algorithm for computing the Hermite normal form described in [DKJ87].

5 Practical Results

In 1991, S. Düllmann [BD91a] needed 10 days on a distributed network of fourteen Sparc1 and SparcSLC computers to compute the class number of the quadratic field with discriminant $-(4 \times 10^{54} + 4)$ (55 digits) using an improved version of Hafner and McCurley's algorithm with large prime variant. On a single SparcUltra1, this should be approximately equivalent to 7 days and 17 hours. Now, using sieving methods, it takes only 8 hours and 53 minutes to obtain this result on a SparcUltra1.

We present our results for the imaginary quadratic case. We were able to compute the class group for the first four of the following discriminants. For the last two, we have so far only been able to compute relation matrices. The number in parenthesis after the discriminant is the number of decimal digits. The class group is presented as $[m_0\ m_2\ \dots m_{s-1}]$, where $Cl(K) \equiv \bigoplus_{0 \le i < s} \mathbb{Z}/m_i\mathbb{Z}$.

Δ_1	$-4 \times F_7 = 4 \times (2^{2^7} + 1)$ (40) $= -1 \times 2^2 \times 59649589127497217 \times 5704689200685129054721$
h	17787144930223461408
Cl	[2 8893572465111730704]
Δ_2	$-(4 \times 10^{54} + 4)$ (55) $= -1 \times 2^2 \times 101 \times 109 \times 9901 \times 153469 \times 999999000001 \times 597795771563/$ 34533866654838281
h	1056175002108254379317829632
Cl	[2 2 2 2 2 33005468815882949353682176]
Δ_3	$-56759029509462061499204078404947821190422701840487390196283$ (59) $= -1 \times 235942923943814840172714410183 \times 2405625418246410575130433/$ 26701
h	34708563502858399116135176220
Cl	[34708563502858399116135176220]
Δ_4	$-(4 \times 10^{64} + 4)$ (65) $= -1 \times 2^2 \times 1265011073 \times 15343168188889137818369 \times 515217525265213/$ 267447869906815873
h	178397819605839608466892693850112
Cl	[4 4 11149863725364975529180793365632]

Δ_5	$-$4695204673552245130677413787157851216622805893433443043 0/ 26971349460603 (70)
h	???
Cl	???

Δ_6	$-(4 \times 10^{74} + 4)$ (75)
h	???
Cl	???

Run-time statistics compiled during these computations are contained in the table below. Here, M is the radius of the sieving interval ($x \in (-M, M)$), "# forms" is the number of forms which were used for sieving, t_L is the is the total CPU time in minutes required to generate the relation matrix, t is the total time required to compute the class group, and t_{old} is an estimation of the time required using the techniques in [BD91b]. The computations were all carried out on a SPARC-ultra computer.

Δ	$\lvert FB \rvert$	M	$forms$	t_L	t	t_{old}
Δ_1	1000	69556	436	5.73	7.95	533.31
Δ_2	4100	333836	5231	108.28	532.24	11098.69
Δ_3	5500	470308	15711	315.79	2338.28	?
Δ_4	7300	628972	25978	855.13	6457.28	?
Δ_5	8800	779084	143678	5007.43	?	?

Note that the 55 digit discriminant here is that computed by Düllmann in [BD91a]. Using large prime variation, we note a considerable speed up for the part of the algorithm finding the relations. The column labelled "% lp" contains the percentage of full relations which were formed form large prime relations.

Δ	$\lvert FB \rvert$	M	$forms$	% lp	t_L
Δ_1	1000	69556	279	22.78	4.81
Δ_2	4100	333836	3081	24.26	38.14
Δ_3	5500	470308	7984	52.05	123.69
Δ_4	7300	628972	9642	49.09	253.72
Δ_5	8800	779084	61825	57.41	1456.79
Δ_6	9500	844988	174068	56.31	4671.23

For the real quadratic case, we present some statistics from computing relation matrices (using large prime variant) for discriminants $\Delta = 10^i + 1$.

i	$\|FB\|$	M	$forms$	% lp	t_L
30	355	22628	90	0.00	0.23
35	650	43756	191	0.57	0.71
40	1150	81572	346	2.05	1.50
45	2500	192772	541	3.26	4.65
50	3000	235652	1075	3.61	10.55
55	4400	363212	3122	15.37	44.36
60	6100	522532	6103	45.91	149.99
65	7600	660332	15909	52.46	528.96
70	8800	764572	47955	55.48	1581.51
75	9600	859412	164547	58.07	5739.84

Finally, we present some statistics for pure cubic extensions $K = \mathbb{Q}(\sqrt[3]{m})$ using a result of H.J. Stender ([Ste77]), which allows one to precompute the regulators of certain special fields. Here, $l(\Delta_K)$ is used to denote the decimal length of the field discriminant, $|FB|$ is the size of the factor base, h is the class number, t_L is the is the total CPU time in seconds (if not indicated otherwise) required to generate the relation matrix and t is the total time required to compute the class group. The computations were all carried out on a Sparc4 computer with 32 MByte RAM.

m	$l(\Delta_K)$	$\|FB\|$	h	t_L	t
2431^3-17	22	100	230666616	7	10
2431^3-13	22	100	204575544	6	9
2431^3-11	21	100	67575654	11	14
2431^3-1	14	100	73143	11	13
2431^3+1	19	100	11803968	10	13
2431^3+11	22	100	208464912	6	9
2431^3+13	22	100	171651312	9	11
2431^3+17	19	100	8682552	9	11
46189^3-17	29	200	528112732248	36	62
46189^3-13	28	200	230653109628	43	61
46189^3-11	29	200	534799522773	39	58
46189^3-1	26	200	84034542636	46	62
46189^3+1	28	200	614506494573	41	57
46189^3+11	25	200	21657374964	41	59
46189^3+13	30	200	1201562002152	36	56
46189^3+17	29	200	439589101464	41	62
4869293^3-17	38	400	7672357980222192	130	6 min
4869293^3-13	40	400	181790041779396270	136	7 min
4869293^3-11	42	400	732630532207386456	172	6.5 min
4869293^3-1	41	400	1842592025672169858	133	6 min
4869293^3+1	41	400	931340490377083533	123	6 min
4869293^3+11	37	400	8165641344661584	143	6 min
4869293^3+13	42	400	1038577938978984723	121	6.5 min
4869293^3+17	42	400	1028253243988827468	118	6.5 min

m	$l(\Delta_K)$	$\|FB\|$	h	t_L	t
$840564439^3 - 17$	55	700	3842073100240397835871905	77 min	115 min
$840564439^3 - 11$	54	700	808279508202073001953848	74 min	107 min
$840564439^3 - 1$	55	700	4813779331073984156486937	43 min	76 min
$840564439^3 + 1$	54	700	1806681492048597457841100	76 min	115 min
$840564439^3 + 13$	55	700	2653130558498568497238285	43 min	83 min
$840564439^3 + 17$	53	700	165032036039112923443113	66 min	102 min

References

[BD91a] J. Buchmann and S. Düllmann. Distributed class group computation. In J. Buchmann, H. Ganzinger, and W.J. Paul, editors, *Informatik – Festschrift aus Anlaß des sechzigsten Geburtstages von Herrn Prof. Dr. G. Hotz*, volume 1 of *Teubner–Texte zur Informatik*, pages 68–81. B. G. Teubner, 1991.

[BD91b] J. Buchmann and S. Düllmann. A probabilistic class group and regulator algorithm and its implementation. In *Computational number theory*, Proc. Colloq., Debrecen/Hung. 1989, pages 53–72, 1991.

[BLP93] J. P. Buhler, H. W. Lenstra, Jr., and C. Pomerance. Factoring integers with the number field sieve. In A. K. Lenstra and H. W. Lenstra, Jr., editors, *The development of the number field sieve*, number 1554 in Lecture Notes in Mathematics, pages 50–94. Springer, 1993.

[BNW] J. Buchmann, S. Neis, and D. Weber. Computing class groups with the NFS. to appear.

[Buc89] J. Buchmann. A subexponential algorithm for the determination of class groups and regulators of algebraic number fields. In *Séminaire de Théorie des Nombres*, pages 27–41, Paris, 1988-89.

[CDyDO93] H. Cohen, F. Diaz y Diaz, and M. Olivier. Calculs de nombres de classes et de régulateurs de corps quadratiques en temps sous-exponentiel. In *Séminaire de Théorie des Nombres*, pages 35–46, Paris, 1993.

[Coh95] H. Cohen. *A course in computational algebraic number theory.* Springer, Heidelberg, 1995.

[DKJ87] P. D. Domich, R. Kannan, and L. E. Trotter Jr. Hermite normal form computation using modular determinant arithmetic. *Mathematics of Operations Research*, 12, 1987.

[HHR93] G. Havas, D. F. Holt, and S. Rees. Recognizing badly presented $\mathbb{Z}$-modules. *Linear Algebra and its Applications*, 192, 1993.

[HM91] J.L. Hafner and K.S. McCurley. Asymptotically fast triangularization of matrices over rings. *SIAM J. Comput.*, 20:1068–1083, 1991.

[HM94a] G. Havas and B. S. Majewski. Hermite normal form computation for integer matrices. Technical Report TR0295, Key Centre for Software Technology, Department of Computer Science, The University of Queensland, 1994.

[HM94b] G. Havas and B. S. Majewski. Integer matrix diagonalization. Technical Report TR0277, Key Centre for Software Technology, Department of Computer Science, The University of Queensland, 1994.

[Jac97] M.J. Jacobson, Jr. Applying sieving to the computation of quadratic class groups. to appear in Math. Comp., 1997.

[PZ89] M. Pohst and H. Zassenhaus. *Algorithmic Algebraic Number Theory.* CUP, 1989.

[Sil87] R. D. Silverman. The multiple polynomial quadratic sieve. *Math. Comp.*, 48:757–780, 1987.

[Ste77] H.-J. Stender. Lösbare Gleichungen $ax^n - by^n = c$ und Grundeinheiten für einige algebraische Zahlkörper vom Grade $n = 3, 4, 6$. *J. reine angew. Math.*, 290:24–62, 1977.

[Web97] D. Weber. *On the computation of discrete logarithms in finite prime fields.* PhD thesis, Universität des Saarlandes, 1997.