MATHEMATICS OF COMPUTATION
Volume 85, Number 300, July 2016, Pages 1861–1873
http://dx.doi.org/10.1090/mcom/3058
Article electronically published on December 7, 2015

QUADRATIC SIEVING

THORSTEN KLEINJUNG

Abstract. We propose an efficient variant for the initialisation step of quadratic sieving, the sieving step of the quadratic sieve and its variants, which is also used in sieving-based algorithms for computing class groups of quadratic fields. As an application we computed the class groups of imaginary quadratic fields with 100-, 110-, 120-, and 130-digit discriminants.

The quadratic sieve integer factoring algorithm [9] and sieving-based algorithms for class group computations of quadratic number fields [3,5,8] have a common step which we call in this article quadratic sieving. In both cases this step is one of the two main computational steps. However, in the quadratic sieve, which is surpassed by the number field sieve [6] for larger integers, quadratic sieving is in practice the main computational step.

Given an integer N and a bound F depending on N, the goal of quadratic sieving is to find sufficiently many $x \in \mathbb{Z}$ such that $x^2 - N$ is F-smooth; i.e., $x^2 - N$ is the product of primes not bigger than F. In the case of a class group computation of $\mathbb{Q}(\sqrt{D})$ the integer N is $k^2 D$ with $k \in \mathbb{Z} \setminus \{0\}$, $k \leq F$, usually $k = 1$ or $k = 2$, and in the case of the quadratic sieve factoring N' it is $N = kN'$ with small $k \in \mathbb{Z} \setminus \{0\}$ chosen depending on N'.

In the quadratic sieve such an x gives rise to the congruence $x^2 \equiv x^2 - N \pmod{N'}$ and multiplying a suitable set of these congruences such that the exponents of the primes on the right hand side become even results in a congruence $y^2 \equiv z^2 \pmod{N'}$ with an at least 50% chance of splitting N' by considering $\gcd(y + z, N')$.

In the context of class group computations such an x implies that the prime ideal decomposition of $(x + k\sqrt{D})$ contains only prime ideals of norm at most F and gives rise to a relation in the ideal class group. Collecting sufficiently many relations and using the generalised Riemann hypothesis allows us to construct a group surjection on the sought class group with small kernel, and starting from this surjection it is easy to compute the class group (cf. Section 3).

After reviewing known variants of quadratic sieving in the first section, a modified approach of the self initialisation variant is described and analysed in the second section and applied to class group computations in the final section.

1. Quadratic sieving

As above we denote by N an integer which is, in the case of the quadratic sieve, a multiple of the number to be factored and, in the case of a class group computation of $\mathbb{Q}(\sqrt{D})$, such that $\mathbb{Q}(\sqrt{N}) = \mathbb{Q}(\sqrt{D})$.

Received by the editor November 21, 2013 and, in revised form, January 6, 2015.
2010 *Mathematics Subject Classification.* Primary 11Y05, 11Y16.

Definition 1.1. Given N and a bound F, a relation is a triple $(x, f, S) \in \mathbb{Z}^3$ such that $x^2 - N = f^2 S$ holds, S is an F-smooth integer, and, in the case of a class group computation, $f = 1$.

Remark 1.2. In the case of a class group computation such a relation produces a decomposition of an element of $\mathbb{Q}(\sqrt{N})$, namely of $(x + \sqrt{N})$, into prime ideals of bounded norm. Since a relation for $4N$ gives a decomposition of an element of the same number field, it is as useful as a relation for N. Similarly, in the case of the quadratic sieve there is no distinction between N and $4N$. Furthermore, the quadratic residuosity of N and of $4N$ modulo an odd prime is the same. Therefore we will not distinguish between relations for N and relations for $4N$ in the following. Moreover, at some places we will assume that N is divisible by 4 if necessary.

First a general description of quadratic sieving is given followed by specialisations to several variants. Let

$$L[\alpha, c] = L_N[\alpha, c] = \exp((c + o(1))(\log N)^{\alpha}(\log\log N)^{1-\alpha})$$

with $o(1)$ for $N \to \infty$ be the usual L-notation. Let $E = 2$ in the case of the quadratic sieve and $E = 3$ in the case of a class group computation, set $\beta = (\frac{1}{\sqrt{2}})^E$, and choose a factor base bound $F = L[\frac{1}{2}, \beta]$ (i.e., $F = L[\frac{1}{2}, \frac{1}{2}]$ or $F = L[\frac{1}{2}, \frac{1}{\sqrt{8}}]$). Denote by $\mathcal{F}$ (the factor base) the set of primes $p \le F$ such that N is a quadratic residue modulo p; for simplicity we always include $p = 2$ in $\mathcal{F}$. Moreover, let $m \ge 1$ be an integer, let $Q_i \in \mathbb{Z}[x]$, $1 \le i \le m$, be quadratic polynomials with discriminant N or $4N$, and let $I_i = [r_i - M, r_i + M] \subset \mathbb{R}$, $1 \le i \le m$, be bounded intervals where $M > 0$ is constant and $r_i \in \mathbb{R}$; these intervals are called sieving intervals. Set $\mathcal{A} = \{(j, y) \in \mathbb{Z}^2 \mid 1 \le j \le m, y \in I_j \cap \mathbb{Z}\}$; this is called the sieving region. The goal of quadratic sieving is to find sufficiently (i.e., about $\#\mathcal{F} \approx \frac{F}{2\log F} = L[\frac{1}{2}, \beta]$) many pairs $(j, y) \in \mathcal{A}$ such that $Q_j(y)$ is F-smooth. With $Q_i = A_i x^2 + B_i x + C_i$ such an F-smooth value gives rise to $4A_j Q_j(y) = (2A_j y + B_j)^2 - \mathrm{disc}(Q_j)$ which is a relation if $4A_j$ is F-smooth (or, in the case of the quadratic sieve, if the square-free part of $4A_j$ is F-smooth).

Finding such pairs can be done using a sieve as follows. For each prime $p \le F$ and each $1 \le j \le m$ one first computes a set $\mathcal{S}_{p,j} \subset \mathbb{Z}$ of minimal cardinality such that the following holds:

$$p \mid Q_j(y) \quad \Leftrightarrow \quad y \in \mathcal{S}_{p,j} + p\mathbb{Z}.$$

For an odd prime p, the set $\mathcal{S}_{p,j}$ is empty if N is a quadratic non-residue modulo p, it is of cardinality one for $p \mid N$, and otherwise it is of cardinality two. Thus only primes in $\mathcal{F}$ need to be considered. A triple (p, j, y) with $p \mid Q_j(y)$ and $(j, y) \in \mathcal{A}$ is called a sieving event, and sorting sieving events with respect to the two last components gives for each (j, y) the set of prime divisors of $Q_j(y)$ not exceeding F, which allows us to decide easily whether $Q_j(y)$ is F-smooth or not. Often the sorting step is combined with the latter step by adding approximations of logarithms of factor base elements to an array indexed by the elements of $\mathcal{A}$ followed by a final inspection of this array.

The cost of this procedure consists of the initialisation cost (computing the sets $\mathcal{S}_{p,j}$) and the sieving cost (generating triples and inspecting the sieving region). Let $t = \#\mathcal{A}$ be the size of the sieving region. Then the expected number of triples (p, j, y) as above is $\sum_{p \in \mathcal{F}} \frac{2}{p} t \approx t \log\log F$. Since this number is bigger (for $F > e^e$) than t, the cost of generating and handling the triples dominates the

cost of inspecting the sieving region and henceforth the latter will be ignored. The costs for initialisation and for sieving depend on the algorithm, and these costs will be compared for fixed F for the variants described in the following.

The size t of the sieving region (i.e., the values m and M) must be chosen such that enough relations are produced. Let X be the maximum of $|Q_j(y)|$ for $(j, y) \in \mathcal{A}$. Heuristically, by the theorem of Canfield-Erdős-Pomerance [4] the smoothness probability for the values $Q_j(y)$ is about $u^{-u+o(1)}$ where $u = \frac{\log X}{\log F}$. In all variants considered below, $X = L[1, \frac{1}{2}]$ holds, which gives $L[\frac{1}{2}, -\frac{1}{2}]$ for the smoothness probability, which in turn implies $t = L[\frac{1}{2}, 1]$.

1.1. **The basic quadratic sieve.** This variant [9] is obtained by setting $m = 1$, $Q_1 = x^2 - N$, $r_1 = \sqrt{N}$, and $M = F^{E+o(1)}$. Splitting the interval I_1 into parts of length $F^{1+o(1)}$, the initialisation cost as well as the sieving cost is $F^{1+o(1)}$ per part. Thus the memory size is $F^{1+o(1)}$ and the total cost is $F^{E+o(1)}$. Furthermore, one gets $X = \sqrt{N}F^{E+o(1)}$ for the maximum of the polynomial values.

1.2. **The multiple polynomial quadratic sieve.** This variant [11] is obtained by setting $M = F^{1+o(1)}$, $m = F^{E-1+o(1)}$, and choosing the quadratic polynomials $Q_i = A_i x^2 + B_i x + C_i$ of discriminant N such that A_i and B_i are of size $\frac{\sqrt{N}}{M}$ and A_i is an F-smooth, odd number for which $x^2 \equiv N \pmod{4A_i}$ has a non-trivial solution (replacing N by $4N$ if necessary; cf. Remark 1.2). Furthermore, set $r_i = -\frac{B_i}{2A_i}$. After precomputing square roots of $N \bmod p$ for each p in the factor base, the initialisation for one polynomial Q_i consists of computing $\mathcal{S}_{p,i}$ for each $p \in \mathcal{F}$, which amounts to reducing A_i and B_i modulo p as well as a few operations modulo p. Therefore, the initialisation cost as well as the sieving cost per polynomial is $F^{1+o(1)}$, which gives a total cost of $F^{E+o(1)}$ using memory of size $F^{1+o(1)}$. The maximum of the polynomial values is $X = \sqrt{N}F^{1+o(1)}$.

Let $u_B = \frac{\log(\sqrt{N}F^{E+o(1)})}{\log F}$ and $u_M = \frac{\log(\sqrt{N}F^{1+o(1)})}{\log F}$ be the values for u in the theorem of Canfield-Erdős-Pomerance for the basic quadratic sieve and the multiple polynomial quadratic sieve, respectively. This gives $u_B = u_M + E - 1 + o(1)$ and for the quotient of the smoothness probabilities

$$\frac{u_M^{-u_M+o(1)}}{u_B^{-u_B+o(1)}} = (eu_M)^{E-1+o(1)} = (\log N)^{\frac{E-1}{2}+o(1)}.$$

Since $M = F^{1+o(1)}$ can be chosen such that the sieving cost dominates the initialisation cost, the sieving cost per element of the sieving region is the same in both variants, and the sieving region can be decreased by the quotient above, the running time is decreased by the same quotient. Therefore, heuristically, the running time of this variant is lower by a factor of $\sqrt{\log N}^{1+o(1)}$ for integer factoring and by a factor of $(\log N)^{1+o(1)}$ for class group computations.

1.3. **The self initialisation variant.** This variant [10] differs from the previous variant in the choice of the leading coefficients A_i. Let $n \in \mathbb{N}$ be such that $m = 2^{n-1} = F^{E-1+o(1)}$ and let $a_1, \ldots, a_n$ be pairwise co-prime odd F-smooth numbers of size at most $\sqrt[n]{\frac{\sqrt{N}}{M}}$ such that N is a non-trivial quadratic residue modulo each a_k. The leading coefficient of each polynomial Q_i is set to the same value $A = A_i = g^2 \prod_{k=1}^{n} a_k$ where g is chosen as an odd F-smooth integer, co-prime

to $N\prod_{k=1}^{n} a_k$, and such that A is of size $\frac{\sqrt{N}}{M}$ and $x^2 \equiv N \pmod{g^2}$ has solutions. Since $x^2 \equiv N \pmod{a_k}$ has at least two solutions, $x^2 \equiv N \pmod{4A}$ has at least $2^n = 2m$ solutions, and discarding one of each pair $\pm x$ gives m values for B_i and thus m quadratic polynomials Q_i. If the a_k are chosen as the first $n = O(\sqrt{\log N \log\log N}) = F^{o(1)}$ primes for which N is a non-trivial quadratic residue, then, heuristically, they will be bounded by $O(n \log n)$. Thus they will asymptotically belong to the factor base and be bounded by $\sqrt[n]{\frac{\sqrt{N}}{M}}$.

After some precomputations like square roots of A modulo each factor base element, the initialisation cost per polynomial and per factor base element consists of one modular addition as opposed to several modular operations and reductions in the multiple polynomial variant. However, asymptotically, this has no effect and the two variants behave similarly.

Remark 1.3. If N is small, there are not enough small primes a_k, and it is not possible to use the same leading coefficient A for all polynomials Q_i. If N is not too small, a smaller n and several leading coefficients A can be used, each being a product of n primes around $\sqrt[n]{\frac{\sqrt{N}}{M}}$ such that N is a non-trivial quadratic residue modulo these primes. Each A gives rise to 2^{n-1} polynomials Q_i for which the initialisation can be done as described above. These considerations apply to numbers N for which quadratic sieving is currently used in practice.

2. Improved initialisation

Lowering the length of the intervals I_i below $F^{1+o(1)}$ is prevented by the separation of the initialisation step and the sieving step. In the following the initialisation step will be modified, which allows us to combine the two steps and to decrease the length of the intervals I_i.

Let the notation be as in Subsection 1.3 and choose B_i as follows. For $k = 2, \ldots, n$ let $B^{(k)} = b_k \frac{A}{a_k}$ with $0 \le b_k < \frac{a_k}{2}$ be such that $(B^{(k)})^2 \equiv N \pmod{a_k}$ holds, and let $B^{(1)} = b_1 \frac{A}{a_1}$ with $0 \le b_1 < a_1$ be such that $(B^{(1)})^2 \equiv N \pmod{a_1}$ and $B^{(1)} + B^{(2)} + \ldots + B^{(n)} \equiv N \pmod 2$ hold. Then the 2^{n-1} solutions (up to sign) of $x^2 \equiv N \pmod{4A}$ are given by $B^{(1)} \pm B^{(2)} \pm \ldots \pm B^{(n)}$. Associate to each $1 \le i \le m = 2^{n-1}$ a different choice of these signs and denote it by $(\epsilon_k^{(i)})$ with $\epsilon_1^{(i)} = 1$. Then set $B_i = \sum_{k=1}^{n} \epsilon_k^{(i)} B^{(k)}$; this satisfies $B_i^2 \equiv N \pmod{4A}$ and $-\frac{n-1}{2}A \le B_i < \frac{n+1}{2}A$.

For primes $p \le 2M$ there is at least one sieving event (p, j, y) for each j; thus initialisation does not dominate the sieving cost for these primes. Hence, the initialisation for these primes can be done in a straightforward way without dominating the cost. For simplicity primes p dividing A will be discarded from the following discussion; the algorithm below can be adapted to these primes (cf. Remark 2.3).

Let $p > 2M$ be a prime in the factor base not dividing A. Then a triple (p, j, y) is a sieving event if and only if the two conditions

$$y \equiv -\frac{B_j - s}{2A} \pmod p \qquad \text{and} \tag{2.1}$$

$$y \in \left[-\frac{B_j}{2A} - M, -\frac{B_j}{2A} + M\right] \tag{2.2}$$

are satisfied where s is a square root of $N \bmod p$. In this case we say that the sieving event is induced by (p, s).

Let $\overline{\frac{B_j}{2A}}$, $\overline{\frac{s}{2A}}$, resp. $\overline{\frac{B^{(k)}}{2A}}$, be representatives in $0, \ldots, p-1$ of $\frac{B_j}{2A} \bmod p$, $\frac{s}{2A} \bmod p$, resp. $\frac{B^{(k)}}{2A} \bmod p$, and denote by $\mathrm{rem}_p(x)$ for $x \in \mathbb{R}$ the unique number in $[0, p[$ such that $x - \mathrm{rem}_p(x)$ is an integer multiple of p. For fixed j there is at most one sieving event (p, j, y) induced by (p, s), and it follows from (2.1) and (2.2) that such an event exists if and only if

$$\mathrm{rem}_p\left(-\overline{\frac{B_j}{2A}} + \overline{\frac{s}{2A}} + \frac{B_j}{2A}\right) \in [0, M] \cup [p - M, p[\tag{2.3}$$

holds.

Since $\mathrm{rem}_p\left(-\overline{\frac{B_j}{2A}} + \frac{B_j}{2A}\right) = \mathrm{rem}_p\left(\sum_{k=1}^{n} \epsilon_k^{(j)}(-\overline{\frac{B^{(k)}}{2A}} + \frac{B^{(k)}}{2A})\right)$, the set of all sieving events (p, j, y) induced by (p, s) can be found by determining all tuples $(\delta_1 = 1, \delta_2, \ldots, \delta_n)$ such that $\mathrm{rem}_p\left(\overline{\frac{s}{2A}} + \sum_{k=1}^{n} \delta_k \beta_k\right)$ lies in $[0, M] \cup [p - M, p[$ where $\beta_k = -\overline{\frac{B^{(k)}}{2A}} + \frac{B^{(k)}}{2A}$. In the self initialisation variant this condition is checked for each tuple (δ_k) individually, leading to $O(2^n)$ operations.

This cost can be reduced by precomputing tables with partial sums as follows. Set $n = n_1 + n_2$ with $n_1 = \lceil \frac{n}{2} \rceil$ and $n_2 = \lfloor \frac{n}{2} \rfloor$, and rewrite

$$\mathrm{rem}_p\left(\overline{\frac{s}{2A}} + \sum_{k=1}^{n} \delta_k \beta_k\right) = \mathrm{rem}_p\left(\mathrm{rem}_p\left(\overline{\frac{s}{2A}} + \sum_{k=1}^{n_1} \delta_k \beta_k\right) - \mathrm{rem}_p\left(-\sum_{k=n_1+1}^{n} \delta_k \beta_k\right)\right).$$

After having sorted the two arrays containing $\{\mathrm{rem}_p(\overline{\frac{s}{2A}} + \sum_{k=1}^{n_1} \delta_k \beta_k)\}$, resp. $\{\mathrm{rem}_p(-\sum_{k=n_1+1}^{n} \delta_k \beta_k)\}$, the sieving events can be found efficiently by comparing appropriate ranges of these arrays while keeping track of the tuples (δ_k). Furthermore, sorting these arrays can be sped up by constructing and sorting them inductively as follows. Let $\mathcal{T}_r$ for $r = 1, \ldots, n_2$ be the sorted array containing the 2^r elements $\mathrm{rem}_p(-\sum_{k=n_1+1}^{n_1+r} \delta_k \beta_k)$ with $\delta_k \in \{-1, +1\}$ and denote by $\mathcal{T}_r + \beta$ an array whose i-th element is $\mathrm{rem}_p(\mathcal{T}_r[i] + \beta)$. Sorting the array $\mathcal{T}_r + \beta$ consists of determining the point where rem_p jumps from p to 0 and moving the upper part of the array to the beginning. Then $\mathcal{T}_{r+1}$ can be computed by sorting $\mathcal{T}_r + \beta_{n_1+r+1}$, sorting $\mathcal{T}_r - \beta_{n_1+r+1}$, and a merge sort of these two sorted arrays. The set $\{\mathrm{rem}_p(\overline{\frac{s}{2A}} + \sum_{k=1}^{n_1} \delta_k \beta_k)\}$ can be handled similarly.

This procedure is summarised in the algorithms below, where the term k-array is used for an array of k-tuples.

Algorithm 2.1 (SHIFT). *Input: a k-array $\mathcal{T}$ of length l which is sorted with respect to the first component, $\beta \in \mathbb{R}$, and $\delta \in \{-1, +1\}$:*

Output: a $(k+1)$-array $\mathcal{T}'$ of length l obtained by adding β modulo p to the first component of each element and sorting with respect to the first component as well as appending δ to each k-tuple:

(1) *Set $\beta' = \mathrm{rem}_p(\beta)$.*
(2) *Append δ to each element of $\mathcal{T}$ and denote this array by $\mathcal{T}_\delta$.*
(3) *Find (e.g., by a binary search) the largest $0 \le z \le l$ such that the first components of the first z elements are smaller than $p - \beta'$.*

(4) *Set*

$$\mathcal{T}'[i] = \begin{cases} \mathcal{T}_\delta[i+z] + (\beta' - p, 0, \ldots, 0) & \text{for } 1 \le i \le l - z \\ \mathcal{T}_\delta[i+z-l] + (\beta', 0, \ldots, 0) & \text{for } l - z < i \le l. \end{cases}$$

Algorithm 2.2 (Main algorithm)**.** *Input: integers N, M, F, n, $A = \prod_{k=1}^n a_k$ where $a_1, \ldots, a_n$ are pairwise co-prime odd F-smooth numbers such that N is a non-trivial quadratic residue modulo each a_k and such that $A \approx \frac{\sqrt{N}}{M}$, a pair (p, s) where $2M < p \le F$ is a prime not dividing A and s is a square root of $N \bmod p$.*

Output: all sieving events induced by (p, s).

(1) *For $k = 2, \ldots, n$ set $B^{(k)} = b_k \frac{A}{a_k}$ with $0 \le b_k < \frac{a_k}{2}$ such that $(B^{(k)})^2 \equiv N \pmod{a_k}$ holds. Then set $B^{(1)} = b_1 \frac{A}{a_1}$ with $0 \le b_1 < a_1$ such that $(B^{(1)})^2 \equiv N \pmod{a_1}$ and $B^{(1)} + B^{(2)} + \ldots + B^{(n)} \equiv N \pmod 2$ hold.*

(2) *Set $\beta_k = -\overline{\frac{B^{(k)}}{2A}} + \frac{B^{(k)}}{2A}$ for $k = 1, \ldots, n$.*

(3) *Set $n_1 = \lceil \frac{n}{2} \rceil$ and $n_2 = \lfloor \frac{n}{2} \rfloor$.*

(4a) *Let $\mathcal{T}_1^{(1)} = \left\{ \left(\mathrm{rem}_p(\overline{\frac{s}{2A}}) + \beta_1, +1 \right) \right\}$ and iteratively for $r = 2, \ldots, n_1$ do the following:*

- *Compute $\mathcal{T}_{r-1}^{(1),+} = \mathrm{SHIFT}(\mathcal{T}_{r-1}^{(1)}, \beta_r, +1)$.*
- *Compute $\mathcal{T}_{r-1}^{(1),-} = \mathrm{SHIFT}(\mathcal{T}_{r-1}^{(1)}, -\beta_r, -1)$.*
- *Do a merge sort (with respect to the first component) of $\mathcal{T}_{r-1}^{(1),+}$ and $\mathcal{T}_{r-1}^{(1),-}$ to obtain the sorted $(r+1)$-array $\mathcal{T}_r^{(1)}$.*

Then set $\mathcal{T}^{(1)} = \mathcal{T}_{n_1}^{(1)}$.

(4b) *Set $\mathcal{T}_0^{(2)} = \{(0)\}$ and iteratively for $r = 1, \ldots, n_2$ do the following:*

- *Compute $\mathcal{T}_{r-1}^{(2),+} = \mathrm{SHIFT}(\mathcal{T}_{r-1}^{(2)}, \beta_{n_1+r}, -1)$.*
- *Compute $\mathcal{T}_{r-1}^{(2),-} = \mathrm{SHIFT}(\mathcal{T}_{r-1}^{(2)}, -\beta_{n_1+r}, +1)$.*
- *Do a merge sort (with respect to the first component) of $\mathcal{T}_{r-1}^{(2),+}$ and $\mathcal{T}_{r-1}^{(2),-}$ to obtain the sorted $(r+1)$-array $\mathcal{T}_r^{(2)}$.*

Then set $\mathcal{T}^{(2)} = \mathcal{T}_{n_2}^{(2)}$.

(5a) *Set $k_0 = 1$. For $i_1 = 1, \ldots, 2^{n_1 - 1}$ do the following: set $t_1 = \mathcal{T}^{(1)}[i_1][1]$, increase k_0 until $k_0 > 2^{n_2}$ or $\mathcal{T}^{(2)}[k_0][1] \ge t_1 - M$, then find the largest k_1 with $k_0 - 1 \le k_1 \le 2^{n_2}$ such that $\mathcal{T}^{(2)}[k_1][1] \le t_1 + M$ and, for each $k_0 \le i_2 \le k_1$, output the sieving event corresponding to $\mathcal{T}^{(1)}[i_1], \mathcal{T}^{(2)}[i_2]$ (i.e., if $\mathcal{T}^{(1)}[i_1] = (t_1, \delta_1, \ldots, \delta_{n_1})$, $\mathcal{T}^{(2)}[i_2] = (t_2, \delta_{n_1+1}, \ldots, \delta_n)$, if j corresponds to (δ_k), and if y is the unique integer satisfying (2.1) and (2.2), output (p, j, y)).*

(5b) *Set $k_0 = 1$. For $i_1 = 1, \ldots, 2^{n_1 - 1}$ do the following: set $t_1 = \mathcal{T}^{(1)}[i_1][1]$, increase k_0 until $k_0 > 2^{n_2}$ or $\mathcal{T}^{(2)}[k_0][1] \ge t_1 + p - M$, and, for each $k_0 \le i_2 \le 2^{n_2}$, output the sieving event corresponding to $\mathcal{T}^{(1)}[i_1], \mathcal{T}^{(2)}[i_2]$.*

(5c) *Set $k_1 = 0$. For $i_1 = 1, \ldots, 2^{n_1 - 1}$ do the following: set $t_1 = \mathcal{T}^{(1)}[i_1][1]$, increase k_1 until $k_1 \ge 2^{n_2}$ or $\mathcal{T}^{(2)}[k_1 + 1][1] > t_1 + M - p$, and, for each $1 \le i_2 \le k_1$, output the sieving event corresponding to $\mathcal{T}^{(1)}[i_1], \mathcal{T}^{(2)}[i_2]$.*

Remark 2.3. The $F^{o(1)}$ prime divisors $p \mid A$, $p > 2M$ can be handled by a slight modification sketched below. It is possible to choose A such that $p^3 \nmid A$ (cf. Subsection 1.3 and Remark 1.3); let $A = p^{e_p} A'$ with $e_p \le 2$ and $p \nmid A'$. Then (2.1) becomes $p^{e_p} y \equiv -\frac{B_j - s}{2A'} \pmod{p^{e_p+1}}$ whereas condition (2.2) remains the same.

Let $\overline{\frac{B^{(k)}}{2A'}}$ denote the representative in $0, \ldots, p^{e_p+1}-1$ of $\frac{B^{(k)}}{2A'} \bmod p^{e_p+1}$ and set $\beta_k = -\frac{1}{p^{e_p}} \cdot \overline{\frac{B^{(k)}}{2A'}} + \frac{B^{(k)}}{2A}$. Then Algorithm 2.2 outputs triples (p, j, y) with $y \in \frac{1}{p^{e_p}}\mathbb{Z}$, which allows us to determine the sieving events induced by (p, s) by checking if $y \in \mathbb{Z}$.

Proposition 2.4. *Let the notation be as in Algorithm 2.2 and let $w_{(p,s)}$ be the number of sieving events induced by (p, s).*

(1) *Algorithm 2.2 needs $O(2^{\frac{n}{2}} + w_{(p,s)})$ operations (arithmetic operations on integers bounded by $O(A)$).*
(2) *Heuristically, the expected value for $w_{(p,s)}$ is $\frac{2^n M}{p}$.*
(3) *Heuristically, the sieving step can be done in $O(2^{\frac{n}{2}}F + 2^n M \log\log F)$ operations when using Algorithm 2.2 for factor base primes $p > 2M$ (cf. Remark 2.3 for $p \mid A$) and otherwise the standard approach. The storage requirement is $O((2^{\frac{n}{2}} + 2^n M \log\log F)F^{o(1)})$.*

Proof. The complexity of Algorithm 2.1 is linear in the input size l since each of its steps is linear in l. Since the cardinality of the array $\mathcal{T}_r^{(i)}$ is 2^{r-1} for $i = 1$ and 2^r for $i = 2$, steps (4a) and (4b) of Algorithm 2.2 each have complexity $O(2^{\frac{n}{2}})$.

In steps (5a) and (5b) k_0 (resp. k_1 in step (5c)) is increased at most $O(2^{\frac{n}{2}})$ times; thus the number of operations in steps (5a) – (5c) is $O(2^{\frac{n}{2}}) + O(w_{(p,s)})$. Note that handling the n signs δ_i while outputting a sieving event has cost $O(1)$ since the bit size of A is at least n. As steps (1) – (3) have negligible cost, this proves claim (1).

Heuristically, the number of integers in the 2^{n-1} intervals I_i is $2^n M$. Note that this is a heuristic since the length $2M$ of the intervals can (and in the case $E = 3$ will asymptotically) be smaller than 1. The probability that a sieving event is induced by (p, s) is $\frac{1}{p}$, again heuristically since $2M$ need not be divisible by p. This proves claim (2).

Summing over all factor base elements while using the standard approach for $p \leq 2M$, the sieving phase can be done in $O(2^{\frac{n}{2}}F + 2^n M \log\log F)$ operations. The size of a single element of the arrays $\mathcal{T}^{(i)}$, $i = 1, 2$, is $O(\log A) = O(F^{o(1)})$, resulting in a size of $O(2^{\frac{n}{2}}F^{o(1)})$ for the two arrays. Approximately $2^n M \log\log F$ sieving events each of size $O(F^{o(1)})$ must be stored. This proves claim (3). □

Setting $m = 2^{n-1} = F^{2E-2+o(1)}$ and $M = F^{2-E+o(1)}$ the total sieving cost is again $F^{E+o(1)}$, whereas the memory requirement increases to $F^{E+o(1)}$ and the maximum of the polynomial values drops to $X = \sqrt{N}F^{2-E+o(1)}$. By the same heuristic argument as in Subsection 1.2 the smoothness probabilities are increased by a factor of $(\log N)^{\frac{E-1}{2}+o(1)}$ compared to the multiple polynomial quadratic sieve.

How this carries over to the running time depends on the model one uses for accessing memory. Additionally, in the case of class group computations the length $2M = F^{-1+o(1)}$ of the intervals is asymptotically smaller than 1; thus the number $m = F^{4+o(1)}$ of quadratic polynomials is much bigger than the size of the sieving region $t = F^{3+o(1)}$. Therefore, the usual implementation techniques such as keeping an array indexed by the sieving region $\mathcal{A}$ and adding for each sieving event (p, j, y) an approximation of $\log p$ to the corresponding array position do not work. One way to overcome this is to store all $t \log\log F$ sieving events and sort them, which will increase the number of memory accesses and the running time by a factor of $\log(t \log\log F) = (\log N)^{\frac{1}{2}+o(1)}$.

Remark 2.5. If $p \nmid s$, which is the case for most primes, one can handle (p, s) and $(p, -s)$ simultaneously by joining their corresponding arrays $\mathcal{T}^{(1)}$.

Remark 2.6. Instead of computing with rational numbers in the first component of the elements of $\mathcal{T}^{(1)}$ and $\mathcal{T}^{(2)}$ (or, after multiplying by $2A$, with bigger integers), one can use floating point numbers with sufficient precision. By relaxing the bounds in (2.3) to accommodate for rounding errors one gets slightly more pairs (i_1, i_2) in steps (5a) – (5c), but it can be checked easily whether the corresponding triple (p, j, y) is a sieving event or not.

Remark 2.7. In practice we are currently in the case $M > n$, which allows some simplifications. Let $p > 2M + \frac{n}{2}$ be a prime with $p \nmid A$, and relax condition (2.2) to

$$y \in \left[-\frac{n+1}{4} - M, \frac{n-1}{4} + M\right]. \tag{2.4}$$

Heuristically, the number of triples (p, j, y) satisfying conditions (2.1) and (2.4) is about $1 + \frac{n}{4M}$ times the number of triples satisfying conditions (2.1) and (2.2), i.e., sieving events. Thus searching for triples satisfying (2.1) and (2.4) followed by a check whether they satisfy (2.2) does not increase the running time significantly for $M > n$. From $p > 2M + \frac{n}{2}$ it follows that for fixed j there is at most one triple (p, j, y) satisfying conditions (2.1) and (2.4), and such a triple exists if and only if

$$\mathrm{rem}_p\left(-\frac{\overline{B_j}}{2A} + \frac{\overline{s}}{2A}\right) \in \left[0, \frac{n-1}{4} + M\right] \cup \left[p - \frac{n+1}{4} - M, p\right[$$

holds. Then the search for such triples can be done by replacing $\mathcal{T}^{(i)}$, $i = 1, 2$, by

$$\mathcal{T}^{(1)} = \left\{ \left(\mathrm{rem}_p\left(\left(-\sum_{k=1}^{n_1} \delta_k \frac{\overline{B^{(k)}}}{2A}\right) + \frac{\overline{s}}{2A}\right), \delta_1, \ldots, \delta_{n_1} \right) \;\middle|\; \delta_1 = 1, \delta_k \in \{-1, +1\} \right\}$$

and

$$\mathcal{T}^{(2)} = \left\{ \left(\mathrm{rem}_p\left(\sum_{k=n_1+1}^{n} \delta_k \frac{\overline{B^{(k)}}}{2A}\right), \delta_{n_1+1}, \ldots, \delta_n \right) \;\middle|\; \delta_k \in \{-1, +1\} \right\}$$

and proceeding as before. Notice that the first components of the elements of these arrays are integers in $[0, p-1]$, thus much easier to handle than the rational numbers in the general version.

Remark 2.8. Remark 1.3 of Subsection 1.3 applies here as well. Furthermore, one can adapt the sizes of $\mathcal{T}^{(1)}$ and $\mathcal{T}^{(2)}$ to the size of the prime p as follows. Let $n' < n$, $n'_1 = \lceil \frac{n'}{2} \rceil$, and the values of $\delta_k \in \{-1, +1\}$ be fixed for $k = n' + 1, \ldots, n$. Then $\mathcal{T}^{(1)}$ is built for all $2^{n'_1 - 1}$ choices of $\delta_1 = 1, \delta_2, \ldots, \delta_{n'_1}$ and $\mathcal{T}^{(2)}$ for all choices of $\delta_{n'_1+1}, \ldots, \delta_{n'}$. Since $\mathcal{T}^{(2)}$ has to be rebuilt for each choice of $\delta_{n'+1}, \ldots, \delta_n$ the cost increases from $O(2^{\frac{n}{2}} + w_{(p,s)})$ to $O(2^{\frac{2n-n'}{2}} + w_{(p,s)})$. As long as $2^{\frac{2n-n'}{2}}$ does not dominate $w_{(p,s)}$ this does not increase the running time significantly, and it has the advantage that the sieving events for these primes are already partially ordered.

Remark 2.9. In the case of integer factoring the maximum of the polynomial values is $X = \sqrt{N} F^{o(1)}$, which is the same order of magnitude as for the continued fraction method, the predecessor of the quadratic sieve (cf. final paragraph of [9]).

3. Some class group computations

As an application the class groups of the imaginary quadratic fields $\mathbb{Q}(\sqrt{D_d})$, $d = 100, 110, 120, 130$, were computed where $D_d = -\lceil 10^{d-1}\pi \rceil - \Delta_d$ with $\Delta_d \geq 0$ being the smallest integer such that D_d is square-free (we have $\Delta_{100} = \Delta_{110} = \Delta_{120} = 1$ and $\Delta_{130} = 0$).

Let $K = \mathbb{Q}(\sqrt{D})$ be an imaginary quadratic field. We used the sieving-based approach for calculating the class group $\mathrm{Cl}(K)$, which is reviewed below, and applied the technique from above for the sieving part. The correctness of this approach depends on the generalised Riemann hypothesis (GRH) since the result of Bach [1] that $\mathrm{Cl}(K)$ is generated by the prime ideals over primes below $B_{Bach} = 6(\log(\mathrm{disc}(K)))^2$ is used. The lower bound of [2] was not used here.

Let $\mathcal{P}$ be the set of prime ideals of $\mathcal{O}_K$ and let $\mathcal{P}'$ be a subset consisting of the ramified prime ideals of norm less than B_{Bach} and for each splitting prime $p < B_{Bach}$ of exactly one of the prime ideals over p (for simplicity we include one prime ideal over 2 in $\mathcal{P}'$). Such a choice of $\mathcal{P}'$ gives a one-to-one correspondence between elements of $\mathcal{P}'$ and primes below B_{Bach} that can appear in a sieving relation. Since primes p decomposing as $(p) = \mathfrak{p}\mathfrak{p}'$ with $\mathfrak{p} \neq \mathfrak{p}'$ imply the relation $[\mathfrak{p}] = -[\mathfrak{p}']$ in $\mathrm{Cl}(K)$ and since prime ideals over inert primes are principal, Bach's result implies (under GRH) that the class group can be written as

$$\mathrm{Cl}(K) \cong \left(\bigoplus_{\mathfrak{p} \in \mathcal{P}'} \mathbb{Z} \right) \Big/ \Lambda'$$

where Λ' is the lattice of all relations among the prime ideals in $\mathcal{P}'$.

A relation in $\mathrm{Cl}(K)$ of the form

$$[\tilde{\mathfrak{p}}] + \sum_{\mathfrak{p} \in \mathcal{P}' \setminus \{\tilde{\mathfrak{p}}\}} a_{\mathfrak{p}} [\mathfrak{p}] = 0 \tag{3.1}$$

allows us to remove $\tilde{\mathfrak{p}}$ from $\mathcal{P}'$ while adjusting Λ' appropriately. Thus, the idea is to produce relations for some set of prime ideals and, using relations of the form (3.1), to extend this set to a set $\mathcal{P}''$ containing $\mathcal{P}'$; let Λ'' be a lattice of relations among the prime ideals in $\mathcal{P}''$. This gives a surjection $\left(\bigoplus_{\mathfrak{p} \in \mathcal{P}''} \mathbb{Z}\right) \Big/ \Lambda'' \twoheadrightarrow \mathrm{Cl}(K)$. If the left hand side is finite, its cardinality is a multiple of the class number; if in addition this cardinality can be factored, the class group can be determined using binary quadratic forms.

In more detail the steps are:

(1) Using quadratic sieving, relations $(x_i, 1, S_i)$ are generated where S_i is F-smooth. In practice, it is advantageous to use a large prime variant, allowing relations with S_i to be L-smooth for some $L > F$. Such a variant differs in the part of sieving that inspects the sieving region and examines $(j, y) \in \mathcal{A}$ if the quotient of $|Q_j(y)|$ by the primes discovered by the sieve is below a bound C. If a quick test reveals that the quotient is L-smooth, the pair (j, y) gives rise to a relation.

As indicated in Remark 1.3, the integers $|D_d|$ are not sufficiently large to allow for a common leading coefficient of the quadratic polynomials. Thus, multiple leading coefficients A_i were used, each a product of n elements from a set of $\tilde{n} > n$ factor base primes near $\sqrt[n]{\frac{\sqrt{N}}{M}}$ (so $g = 1$).

(2) After removing duplicate relations and repeatedly removing relations containing a prime that appears in no other relation (singleton removal), the relations are converted into an overdetermined matrix over $\mathbb{Z}$ in which columns correspond to relations and rows correspond to primes. Elementary column operations as well as removing certain rows and columns are used to reduce the size of the matrix while keeping it sparse and slightly overdetermined (filtering step) and to produce a smaller matrix denoted by R. By making a choice for splitting primes, coinciding with the choice made for $\mathcal{P}'$, a set $\mathcal{F}'$ of prime ideals can be chosen whose elements correspond to the rows of R.

(3) A tentative multiple $\tilde{h}$ of the class number h together with its prime factorisation is generated. After dropping random columns from R to produce a square matrix R', the (block) Wiedemann algorithm and the Chinese remainder theorem are used to compute $\det(R')$ as described in [8]. If the greatest common divisor of several such non-zero determinants is $\sqrt{N}^{1+o(1)}$ (or below N), it can be used for $\tilde{h}$ since the time for obtaining its factorisation is negligible.

In practice we did not encounter an example where it was necessary to calculate more than two determinants, and the determinants were always non-zero. Notice that, asymptotically, it is sufficient to compute just one non-zero determinant since one can recover its prime factors up to $\sqrt{N}^{1+o(1)}$ by the elliptic curve method (ECM) [7] in heuristic running time $L[\frac{1}{2}, 1]$. In the four class group computations for $\mathbb{Q}(\sqrt{D_d})$ we made a certain ECM effort after the computation of the first determinant and were able to skip the second determinant computation in two cases.

(4) The exponent h' of $\mathrm{Cl}(K)$, i.e., the smallest $h' > 0$ such that exponentiation by h' annihilates $\mathrm{Cl}(K)$, is determined by calculating the order of all prime ideals in $\mathcal{P}'$ using the prime factors of $\tilde{h}$ as a hint. If this step fails, the tentative $\tilde{h}$ from the previous step is not a multiple of h and one has to go back to the previous step. This did not happen in the four class group computations.

(5) For each prime p dividing h' the p-Sylow group $\mathrm{Cl}(K)_p$ of $\mathrm{Cl}(K)$ is calculated as well as the images in $\mathrm{Cl}(K)_p$ of all prime ideals in $\mathcal{F}'$. If p is small, this can be done by ad hoc methods. For larger p the Lanczos algorithm can be used. This is straightforward in the case that $p^2 \nmid h$ and that p is much bigger than the dimension of the matrix R; otherwise the solution space must be processed more carefully, the Lanczos algorithm must be used several times, or the block Lanczos version must be used.

These local data are combined into a certificate consisting of a set of generators and, for each prime ideal in $\mathcal{P}'$, its representation in terms of the generators. If $\mathcal{F}'$ does not contain $\mathcal{P}'$, this is done by first computing representations for the prime ideals in $\mathcal{F}'$ and then, via equation (3.1), extending to other prime ideals using relations from step (1) or, if need be, from additional sieving.

(6) The certificate is verified (under GRH) by checking that the generators are independent, that they have the specified orders, and that each prime ideal in $\mathcal{P}'$ can be expressed in terms of the generators as stated in the certificate.

d	sieving time	Wie1	ECM	Wie2	DL, etc.	total
100	40h	6.6h	0.5h	5.3h	1.5h	54h
110	15d	2d	0.7d	-	0.3d	18d
120	161d	17d	9d	24d	0.4d	210d
130	2.23a	0.36a	0.016a	-	0.02a	2.63a

FIGURE 1. Time for the main computational steps of the class group computations for imaginary quadratic fields with d-digit discriminants. Time is displayed in hours (h), days (d), or years (a).

sieving time	number of columns of R	weight of R	Wiedemann time
1.34a	180 147	19 002 590	4.5a
1.56a	94 427	16 387 161	0.98a
1.78a	81 357	14 760 993	0.66a
2.01a	72 930	13 901 065	0.5a
2.23a	68 108	13 134 334	0.36a

FIGURE 2. Effect of oversieving on the matrix size, the matrix weight, and the time for one determinant computation in step (3). Time is displayed in years (a).

The main computational steps are (1) and (3). Figure 1 shows the timings for these steps, splitting step (3) into the first block Wiedemann (Wie1), the ECM attempt (ECM), and, if need be, the second block Wiedemann (Wie2). The remaining steps are summarised as (DL, etc.) since the Lanczos calculations in step (5) dominate. The times are in core hours (h), core days (d), or core years (a) using idle time on a dual 12-core Opteron at 1.9 GHz. Contrary to the Wiedemann times (see columns Wie1 and Wie2) sieving times were almost unaffected by other jobs. For $d = 120$ the ECM effort is oversized.

Although it is sufficient to generate so many relations that an overdetermined matrix can be produced at the end of step (2), it is usually more efficient to increase the sieving effort in order to reduce the time spent in step (3) (Wie1, ECM, and Wie2). For $d = 130$ about 60% of the relations were sufficient, but the total time for sieving and one block Wiedemann run would have been higher. In Figure 2 the decrease in the matrix size by generating more relations is illustrated for $d = 130$. The final line corresponds to the actual calculation; for the other lines the column "Wiedemann time" is an extrapolation based on a block Wiedemann run modulo a few primes. If it were known beforehand that ECM finds all prime factors of the class number, less sieving could have reduced the total running time.

Some parameters used for the sieving part as well as sieving related data are given in Figure 3. The variables have the same meaning as in the previous sections, namely, n is the number of factors a_i in the leading coefficients of the quadratic polynomials, $\#A$ is the number of different leading coefficients, M is half of the length of the sieving intervals, F is the factor base bound used for sieving, whereas L is the smoothness bound for the large prime variant as explained above, C is the bound for inspecting cofactors, and $t = (\#A) \cdot 2M \cdot 2^{n-1}$ is the total size of the

d	n	a_i-range	$\#A$	M	F	L	C	t	# relations
100	14	$[1693, 2081]$	240 000	2^{13}	200 000	2^{32}	2^{64}	$32.2 \cdot 10^{12}$	2 233 420
110	15	$[2357, 2741]$	1 440 000	2^{12}	400 000	2^{33}	2^{70}	$193 \cdot 10^{12}$	4 952 949
120	16	$[2971, 3463]$	6 360 000	2^{12}	800 000	2^{34}	2^{80}	$1.71 \cdot 10^{15}$	22 314 706
130	18	$[2371, 2801]$	11 245 000	2^{11}	1 600 000	2^{35}	2^{88}	$6.04 \cdot 10^{15}$	72 943 793

FIGURE 3. Parameters (see text) used in the class group computations.

sieving region. The bounds for L and C were chosen such that the running time was increased by at most 1% compared to the variant without large primes, i.e., $L = F$, $C = 1$. The length $2M$ of the sieving intervals was chosen as a power of two (for implementation reasons) and such that the number of relations per time was maximal. Although this length can be chosen asymptotically as $F^{-1+o(1)}$ it is much bigger than 1 in the four cases. One possible reason for this is that the classical approach is very fast if the sieving interval fits into the L1-cache and fast arithmetic modulo p is available, which is the case for, say, $2M \leq 2^{16}$ (L1-cache) and $p \leq 2^{16}$ (SIMD instructions). Another possible reason is that the overhead for the increased number of memory accesses and their less regular pattern reduces the efficiency of the new approach for small N.

Finally, the results of the computations are displayed in Figure 4.

d	factorisation of h and structure of $\mathrm{Cl}(K)$
100	$2^4 \cdot 23 \cdot 523 \cdot 23057 \cdot 25117 \cdot 136256902553 \cdot 4348748846644437283908557$
	$(\mathbb{Z}/2\mathbb{Z}) \times (\mathbb{Z}/3302272559631385170211556966101649164358686831536 8\mathbb{Z})$
110	$2^8 \cdot 11 \cdot 6883 \cdot 15121 \cdot 5107831 \cdot 160104786263 \cdot 5555021823796725692587001$
	$(\mathbb{Z}/2\mathbb{Z})^5 \times (\mathbb{Z}/4160711078960662894140298762727265070260777690117575 2\mathbb{Z})$
120	$2^7 \cdot 7 \cdot 379 \cdot 24133 \cdot 357600656819567292169975729269116269753399329369 07$
	$(\mathbb{Z}/2\mathbb{Z})^5 \times (\mathbb{Z}/9158131222069446450608631360048019813761863719822388808172\mathbb{Z})$
130	$2^9 \cdot 3 \cdot 5 \cdot 26984470254026794913 \cdot 37322833896097955713 \cdot 5947362367424588311471$
	$(\mathbb{Z}/2\mathbb{Z})^4 \times (\mathbb{Z}/28751078900352090638646670385697040209822000202468306326683515 20\mathbb{Z})$

FIGURE 4. Class number and structure of the class group of the imaginary quadratic field $K = \mathbb{Q}(\sqrt{D_d})$ for $d = 100, 110, 120, 130$ where $D_d = -\lceil 10^{d-1}\pi \rceil - \Delta_d$ and $\Delta_{100} = \Delta_{110} = \Delta_{120} = 1$ and $\Delta_{130} = 0$.

REFERENCES

[1] E. Bach, *Explicit bounds for primality testing and related problems*, Math. Comp. **55** (1990), no. 191, 355–380, DOI 10.2307/2008811. MR1023756 (91m:11096)

[2] K. Belabas, F. Diaz y Diaz, and E. Friedman, *Small generators of the ideal class group*, Math. Comp. **77** (2008), no. 262, 1185–1197, DOI 10.1090/S0025-5718-07-02003-0. MR2373197 (2009c:11179)

[3] J.-F. Biasse, *Improvements in the computation of ideal class groups of imaginary quadratic number fields*, Adv. Math. Commun. **4** (2010), no. 2, 141–154, DOI 10.3934/amc.2010.4.141. MR2654130 (2011e:11192)

[4] E. R. Canfield, P. Erdős, and C. Pomerance, *On a problem of Oppenheim concerning "factorisatio numerorum"*, J. Number Theory **17** (1983), no. 1, 1–28, DOI 10.1016/0022-314X(83)90002-1. MR712964 (85j:11012)

[5] M. J. Jacobson Jr., *Applying sieving to the computation of quadratic class groups*, Math. Comp. **68** (1999), no. 226, 859–867, DOI 10.1090/S0025-5718-99-01003-0. MR1604324 (99i:11120)

[6] A. K. Lenstra and H. W. Lenstra, Jr. (eds.), *The Development of the Number Field Sieve*, Lecture Notes in Math. **1554**, Springer, 1993.

[7] H. W. Lenstra Jr., *Factoring integers with elliptic curves*, Ann. of Math. (2) **126** (1987), no. 3, 649–673, DOI 10.2307/1971363. MR916721 (89g:11125)

[8] K. S. McCurley, *Cryptographic key distribution and computation in class groups*, Number theory and applications (Banff, AB, 1988), NATO Adv. Sci. Inst. Ser. C Math. Phys. Sci., vol. 265, Kluwer Acad. Publ., Dordrecht, 1989, pp. 459–479. MR1123090 (92e:11149)

[9] C. Pomerance, *Analysis and comparison of some integer factoring algorithms*, Computational methods in number theory, Part I, Math. Centre Tracts, vol. 154, Math. Centrum, Amsterdam, 1982, pp. 89–139. MR700260 (84i:10005)

[10] C. Pomerance, J. W. Smith, and R. Tuler, *A pipeline architecture for factoring large integers with the quadratic sieve algorithm*, SIAM J. Comput. **17** (1988), no. 2, 387–403, DOI 10.1137/0217023. Special issue on cryptography. MR935347 (89f:11168)

[11] R. D. Silverman, *The multiple polynomial quadratic sieve*, Math. Comp. **48** (1987), no. 177, 329–339, DOI 10.2307/2007894. MR866119 (88c:11079)

Laboratory for Cryptologic Algorithms, Station 14, EPFL, CH-1015 Lausanne, Switzerland