Some algorithms in algebraic number theory

B.Sury

Stat-Math Unit
Indian Statistical Institute
Bangalore
sury@ns.isibang.ac.in

February 2006 - I.I.Sc.

Introduction

These talks address some of the topics mentioned in the survey article by H.W.Lenstra Jr. in the Bulletin of the AMS, 1992. The point of view is that of a number-theorist approaching for the first time the standard results of the subject with a view to looking for constructive proofs. Of course, that is too loose a term; more precisely, one looks for algorithms which can be practically carried out in reasonable time if possible. As the subject has grown in leaps and bounds in the last 15 years (mainly due to its relevance to public key cryptography), it is imperative that the number-theorist introduce herself to these algorithmic aspects at the earliest. At the same time, the aim will be modest in that one is not bothered too much about the most efficient method in terms of complexity.

Let K be an algebraic number field, and let $\mathcal{O}_K$ denote its ring of integers. Suppose $K = \mathbf{Q}(\alpha)$ be given, in the sense that the minimal polynomial of α over $\mathbf{Q}$ is given. At the first level, the principal objects of computation in algebraic number theory are (their definitions will be recalled by and by) :
(i) Factors of a given polynomial $f \in K[X]$.
(ii) An integral basis of K (and hence, the ring $\mathcal{O}_K$).
(iii) The group of units $\mathcal{O}_K^*$.
(iv) The discriminant of K.
(v) The regulator of K.
(vi) The class number of K.
(vii) The class group of K.
(viii) A system of fundamental units of K.
(ix) The Galois group of any (finite) Galois extension of K.
(x) Deciding whether a given polynomial over K is solvable by radicals.
There are several other objects of study which could be discussed at a second level. For example, the ray class fields and various invariants associated to objects like elliptic curves over number fields can be studied. We do not discuss them here. However, in order to discuss any of the ten objectives outlined above, one has to recall some basic algorithms in elementary number theory which are needed in almost all of them. In order to understand the algorithms to compute any of the advanced objects like class group, unit group etc., these will be discussed first.

§ 1. **Classical algorithms in elementary number theory**

1.1 *Power map on a group*
If G is a (usually finite abelian) group, then the map $g \mapsto g^n$ requires at the most $2log|n| + 1$ steps. This is done by writing the base 2 expansion of n and squaring repeatedly.

1.2 *Finding GCD's of integers*
Here, we mean the extended version, where we are given natural numbers a, b and want to compute two integers u, v such that $au + bv$ is the GCD of a and b. Of course, the usual Euclidean algorithm gives us u, v also. If the numbers a, b are at most n, the number of steps of the algorithm is of the order of $log(n)$. As each division takes $O(log(n)^2)$ time, the running time is $O(log(n)^3)$. However, a cleverer way of carrying out the division is due to D.H.Lehmer and the overall running time is only $O(log(n)^2)$.
Note that the Euclidean algorithm provides an algorithm to find the inverse of any element of the group $\mathbf{Z}_m^*$ of all positive integers not exceeding m and coprime to m.

1.3 *GCD's of polynomials*
Although the Euclidean algorithm is valid for polynomials over fields also, it does not make much sense when the polynomials have real or complex coefficients as the coefficients are represented only approximately on a computer. However, if the coefficients are rational numbers or integers, this problem is overcome but there could be a different problem with the coefficients being too large in size. This is because the sum of two rational numbers have numerator and denominator of the magnitude of the product. However, as we shall see later, computations with rational polynomials can often be reduced to those on integral polynomials and the latter can be reduced to computations with polynomials over finite fields. Over finite fields, the Euclidean algorithm works very well.

1.4 *Chinese remainder theorem*
Although this is mathematically as easy to prove for any commutative ring as for the integers, a nice algorithm can be given only for Euclidean rings.
If $m_1, m_2, \cdots, m_r$ are pairwise coprime, and if $a_1, a_2, \cdots, a_r$ are arbitrary integers, a common solution x (which is unique modulo multiples of $m_1 m_2 \cdots m_r$) for all the congruences $x \equiv a_i$ modulo $m_i; 1 \leq i \leq r$ is given as $x =$

$\sum_{i=1}^{r} a_i m_i'(\prod_{j\neq i} m_j)$ where $m_i'(\prod_{j\neq i} m_j) \equiv 1 \bmod m_i$.
Note that we already know from the (extended) Euclidean algorithm how to compute the various m_i'. For this reason, Chinese remainder theorem is easy to apply computationally only when Euclidean algorithm can be applied.

1.5 *Primitive roots*
Many public key cryptosystems require a cyclic group with the knowledge of a generator. Typically, one uses the multiplicative group of a finite field and the simplest example is $\mathbf{Z}_p^*$ for a prime p. In general, one knows that the group $\mathbf{Z}_m^*$ is a cyclic group if, and only if, $m = 2, 4, p^r, 2p^r$ for some odd prime p. A generator of $\mathbf{Z}_m^*$ is called a primitive root mod m.
Finding a primitive root mod p for any prime p, is not an easy problem computationally if p is large. The only method known is to first find all primes q dividing $p-1$ (more or less equivalent to factorising $p-1$), and checking the numbers a from 2 to $p-1$ one by one as to whether $a^{\frac{p-1}{q}}$ is 1 for any prime $q/(p-1)$. A primitive root is found when an a is found for which $a^{\frac{p-1}{q}} \neq 1$ in $\mathbf{Z}_p^*$ for any prime $q/(p-1)$.
However, once a primitive root mod p is found for an odd p, it is easy to get a natural number a which is a primitive root mod p^r for every r. In fact, either a or $a+p$ does the job for each p^r. Finally, if a is a primitive root mod p^r, then the odd number among a and $a+p^r$ is a primitive root mod $2p^r$.

1.6 *Square roots mod primes*
This is a special case of finding a root mod p of an integral polynomial, and can be carried out using the quadratic reciprocity law :
If a, b are odd and $b > 0$, then $(a/b) = (-1)^{(a-1)(b-1)/4}(b/|a|)$.
This tells us how to check if a is a square mod p but how does one find a square root b of a mod p ?
If $p \equiv 3 \bmod 4$, then $b = a^{(p+1)/4}$ works.
If $p \equiv 5 \bmod 8$, then either $b = a^{(p+3)/8}$ (if $a^{(p-1)/4} \equiv 1$) or $(2a)(4a)^{(p-5)/8}$ (if $a^{(p-1)/4} \equiv -1$) works.
The hard case is $p \equiv 1 \bmod 8$. Here, one has the following probabilistic algorithm due to Tonelli and Shanks. This can be made deterministic but becomes exponential time. The algorithm is polynomial time if one assumes the truth of the generalized Riemann hypothesis (GRH) which predicts that all nontrivial zeroes of the Dedekind zeta functions of number fields lie on the line $Re(s) = 1/2$.

Let us describe the algorithm briefly.
Suppose we have found a generator x of a 2-Sylow subgroup P of $\mathbf{Z}_p^*$. Write $p-1 = 2^e q$ with q odd. Now, a^q is a square in P, as $(a^q)^{2^{e-1}} = a^{(p-1)/2} = 1$. Thus, $a^q = x^{2n}$ (as squares in P are just the even powers of x). Hence, $b = a^{(q+1)/2}x^{-n}$ is a solution of $b^2 = a$ in $\mathbf{Z}_p^*$.
Now, the probability of finding a generator x for P is attacked probabilistically as follows. Starting with a random r, one computes $x = r^q \bmod p$. This x generates P if, and only if, $(r/p) = -1$. The probability of finding a quadratic non-residue r is $(p-1)/2p$ which is close to $1/2$ when p is large. Evidently, if one goes through all the elements of $\mathbf{Z}_p^*$ in order to find a quadratic non-residue, one may have to pass over $(p-1)/2$ elements before hitting one, and, thus the deterministic algorithm would take exponential time. Assuming the GRH, one can show that the smallest quadratic non-residue is of the order of $log(p)^2$. This is exactly what is used in the Miller-Rabin primality test too !

1.7 *Finding a root of $f \in \mathbf{F}_p[X]$*
Here, we are dealing with the easy problem of finding roots in $\mathbf{F}_p$ itself. Later, we will see how to factor in $\mathbf{F}_p[X]$. We take p be an odd prime. The idea is that $\mathbf{F}_p$ being the fixed field of the Frobenius map $\phi_p : x \mapsto x^p$ on the algebraic closure of $\mathbf{F}_p$, one has

$$g := GCD(X^p - X, f) = \; c \prod_{\alpha \in \mathbf{F}_p, f(\alpha)=0} (X - \alpha).$$

Then, the algorithm proceeds as follows :
Step 1 : If $g(0) = 0$, replace g by $g(X)/X$ etc. and proceed.
Step 2 : If $g = a_0 + a_1 X$ with $a_0 a_1 \neq 0$, then the only nonzero root of f in $\mathbf{F}_p$ is $-a_0 a_1^{-1}$.
Step 3 : If $g = a_0 + a_1 X + a_2 X^2$, find the square root of the discriminant $a_1^2 - 4a_0 a_2$ by 1.6 say. Then, one can determine the roots of g.
Step 4 : In general, take a random $a \in \mathbf{F}_p$, and find

$$h := GCD((X + a)^{(p-1)/2} - 1, g).$$

If h is either a constant or a constant multiple of g, go to the next a. Note that for a root α of g, $X - \alpha$ is a factor of h exactly in the case when $\alpha + a$ is a quadratic residue mod p. Thus, the chances of *not* finding a proper factor h (that is a factor where both h and g/h are non-constant) is at the most

$1/2^{deg(g)}$.
Step 5 : Write out g/h and h and stop.
Step 6 : Start again with the smaller degree polynomials g/h and h in place of g.

§ 2. **Linear algebra algorithms**

2.1 *Characteristic polynomial and adjoint*
Firstly, one can use Gauss elimination to compute det A for a given $n \times n$ matrix A. There are also variants due to Lewis Carroll and Bareiss. Of course, for a matrix with real or complex entries, one has many numerical methods. If A has integer entries, one could compute the determinant modulo several primes and use the Chinese remainder theorem after having estimated how large $|det(A)|$ is. Of course, the Gaussian elimination method works well over finite fields. Being able to compute determinants, how does one compute the characteristic polynomial ?
A method is to use Lagrange interpolation. One takes $n+1$ distinct points, say $0, 1, \cdots, n$ and finds the values $det(rI - A)$ for $0 \leq r \leq n$. Then, the characteristic polynomial

$$\chi_A(X) := det(XI - A) = \sum_{r=0}^{n} det(rI - A) \prod_{s \neq r} \frac{X - s}{r - s}.$$

Here is a more direct method to compute $\chi_A(X)$ which is based on the statement :

$$\chi'_A(X) = Tr(adj(XI - A)).$$

The proof of the above statement is as follows. Expand $det(XI - A)$ along the i-th column and appeal to the multilinearity of determinant to conclude $\chi'_A(X) = \sum_{i=1}^{n} a_{ii}(X)$, where $a_{ii}(X)$ is the subdeterminant of the $(n-1) \times (n-1)$ matrix obtained by removing the i-th row and the i-th column from $XI - A$.
The algorithm to obtain the characteristic polynomial as well as the adjoint of A can be described as follows :
Start with $a_0 = 1, B_0 = I$. Compute $a_1, \cdots, a_n$ and $B_1, \cdots B_{n-1}$ recursively using

$$a_i = \frac{-Tr(AB_{i-1})}{i} \ , \ B_i = AB_{i-1} + a_i I.$$

Then, $\chi_A(X) = \sum_{i=0}^{n} a_i X^{n-i}$ and $adj(A) = (-1)^{n-1} B_{n-1}$.
The proof that this algorithm is correct goes as follows.
Write $adj(XI - A) = \sum_{i=0}^{n-1} B_i X^{n-i-1}$ and $\chi_A(X) = \sum_{i=0}^{n} a_i X^{n-i}$. Then, the above observation gives

$$(n-i)a_i = Tr(B_i) \ \forall \ i < n.$$

Moreover, the equality $\chi_A(X)I = (XI - A)adj(XI - A)$ gives on comparison of coefficients of like powers of X that

$$a_i I = B_i - AB_{i-1} \ , n > i \geq 1 \ , a_n I = -AB_{n-1} \ , a_0 = 1.$$

Take traces to get $Tr(B_i) = Tr(AB_{i-1}) + na_i$ and we get

$$a_i = \frac{-Tr(AB_{i-1})}{i} \ \forall \ 1 \leq i \leq n.$$

Finally, $B_{n-1} = adj(XI - A)$ at $X = 0 = adj(-A) = (-1)^{n-1}adj(A)$.

2.2 *Free abelian groups defined by an integral matrix*
Most of the groups appearing in algebraic number theory like the class group, the unit group of a number field K, the additive group of any ideal of $\mathcal{O}$ etc. are all finitely generated abelian groups. They are usually given as images, kernels and cokernels of the homomorphisms on free abelian groups defined by integral matrices.
Working with algorithms in linear algebra over the field $\mathbf{Q}$ is insufficient to distinguish between different free abelian subgroups of $\mathbf{Q}^n$ which generate the same $\mathbf{Q}$-vector space.
The best way to work with an integral matrix is to reduce it to a convenient normal form. The two forms commonly used are the Hermite normal form and the Smith normal form.
Hermite normal form (HNF) :
Let $A \in M_{m,n}(\mathbf{Z})$ where $m \leq n$, say. Then, there is a (not necessarily unique) matrix $M \in GL(n, \mathbf{Z})$ such that $B := AM$ is in Hermite normal form. The Hermite normal form of A is unique.
The statement that B is in HNF means :
(i) there is $r \leq n$ for which the first r columns of B are 0, (ii) on the j-th column, if $l(j)$ is maximal such that $b_{l(j),j} \neq 0$, then $b_{l(j),j} \geq 1$, (iii) $l(j+1) > l(j)$, and (iii) all entries to the right of $b_{l(j),j}$ are strictly less but non-negative.
If A has maximal rank (must therefore be m), then $r = n - m$ and the last m columns are linearly independent and form an upper triangular matrix of a special form.
In general, we have :
The image of A (considered as a homomorphism from $\mathbf{Z}^n$ to $\mathbf{Z}^m$) is free abelian and has as a $\mathbf{Z}$-basis the non-zero columns of B.

The kernel of A is free abelian with a $\mathbf{Z}$-basis the first r columns of (any) M as above, where r is the largest number for which the first r columns of B are zero.

The proof of the first statement is clear. Let us prove the second one now. If M_i denotes the i-th column of M, then AM_i is the i-th column of $AM = B$. Therefore, $AM_i = 0$ for $i \leq r$. Solving the system of linear equations $BY = 0$ from the bottom, it follows that the last $n-r$ entries of Y are zero and the first r entries are arbitrary. In other words, the first r canonical vectors $e_1, \cdots, e_r$ of $\mathbf{Z}^n$ form a $\mathbf{Z}$-basis of Ker B. Thus, the vectors $Me_1, Me_2, \cdots, Me_r$ form a $\mathbf{Z}$-basis of Ker A.

A *caveat* is that even with a 20×20 matrix with integral entries ≤ 10, the algorithm to computate the HNF may involve integers with 1500 digits ! Here, we are talking about the simple algrithm similar to using Gaussian elimination to find pivots etc. One way of overcoming these problems is by using the path-breaking LLL-algorithm which we shall talk about shortly.

An application of HNF - to check equality of lattices :

If L, L' are lattices in $\mathbf{Q}^n$ which have the same rank m, then one can use the HNF to check efficiently whether $L = L'$. Indeed, let $d_L, d_{L'}$ denote the denominators of L, L' resectively; these are the smallest natural numbers for which $d_L L, d_{L'} L' \subset \mathbf{Z}^m$. The lattices $d_L L$ and $d_{L'} L'$ have HNF's B, B' say. Then $L = L'$ if, and only if, $B = B'$ and $d_L = d_{L'}$.

2.3 *Finite abelian groups*

Groups like the class group arise as $\mathbf{Z}^n/L$ for some lattice L of maximal rank. The earlier method works for free abelian groups but to find quotients, it is efficient to use the Smith normal form. This is nothing but the matrix given by the elementary divisor theorem. The standard proof in any algebra textbook can also be given as an algorithm. However, both the SNF and the HNF are computed efficiently using the LLL which we shall talk about next.

2.4 *The LLL-algorithm*

This 1982 method due to A.K.Lenstra, H.W.Lenstra Jr. and L.Lovasz broke new ground and has proved a most influential method for computations in number theory - especially in factorisation of polynomials over number fields. In simple terms, this method starts with a basis of a lattice and reduces it to a basis which is nearly orthogonal and whose vectors are 'shorter' in a sense. This reduction is managed by the LLL-algorithm in polynomial time.

In some sense, the LLL method unwarps a badly warped basis. Let us be more precise now.
The set-up is as follows. We consider pairs $(L.q)$ where L is a lattice of rank n and q is a positive-definite quadratic form on the real space $L \otimes \mathbf{R}$. One may define an equivalence $(L, q) \sim (L', q')$ if there is an abelian group isomorphism between the lattices which respects the forms. As a positive-definite quadratic form on $\mathbf{R}^n$ gives rise naturally to a positive-definite symmetric matrix, the equivalence above can be expressed in terms of matrices as follows.
The equivalence classes (L, q) correspond bijectively with the classes of positive-definite symmetric matrices Q, where $Q \sim Q'$ if $Q' = {}^tMQM$ for some $M \in GL(n, \mathbf{Z})$.
Let $\{v_1, \cdots, v_n\}$ be a basis of $\mathbf{R}^n$. Consider the lattice L with this as $\mathbf{Z}$-basis. One calls the positive real number $|det(v_1, \cdots, v_n)|$ given by the absolute value of the determinant of the matrix with v_i's as columns to be the discriminant of L and denotes it by $disc(L)$. Note that a change of $\mathbf{Z}$-basis does not affect the discriminant as the determinant inside can change by ± 1 only. Now, the Gram-Schmidt process produces an orthogonal (not necessarily orthonormal) basis of V in the usual way :
$w_1 = v_1, w_i = v_i - \sum_{j<i} \mu_{ij} w_j$ where $\mu_{ij} = \frac{<v_i, w_j>}{<w_j, w_j>}$.
One defines the $\mathbf{Z}$-basis $\{v_1, \cdots, v_n\}$ of L to be *LLL-reduced* if :
(i) $|\mu_{ij}| \leq \frac{1}{2}$ for all $i > j$, and
(ii) $|w_i + \mu_{i,i-1} w_{i-1}|^2 \geq \frac{3}{4} |w_{i-1}|^2$ for all $i > 1$.
In what follows, the constant $\frac{3}{4}$ in (ii) can be replaced by any $t \in (\frac{1}{4}, 1)$. Note that (ii) is equivalent to $|w_i|^2 \geq (\frac{3}{4} - \mu_{i,i-1}^2) |w_{i-1}|^2$ for $i > 1$ and that the vectors $w_i + \mu_{i,i-1} w_{i-1}$ and w_{i-1} are the projections of v_i and v_{i-1} respectively, on the orthogonal complement of $\sum_{j<i-1} \mathbf{R} v_j$.
Proposition.
Let $\{v_1, \cdots, v_n\}$ be an LLL-reduced basis of L. With w_i's defined as above, we have :
(a) $|v_j|^2 \leq 2^{i-1} |w_i|^2$ *for* $j \leq i$,
(b) $disc(L) \leq \prod_i |v_i| \leq 2^{n(n-1)/4} disc(L)$,
(c) $|v_1| \leq 2^{(n-1)/4} disc(L)^{1/n}$, *and*
(d) *For* $0 \neq x \in L$, $|v_1| \leq 2^{(n-1)/2} max(|x|$.
If the constant $\frac{3}{4}$ in (ii) is replaced by some $t \in (\frac{1}{4}, 1)$, then all the powers of 2 in the proposition are replaced by the same powers of the number $\frac{4}{4t-1}$. Also,

the inequality $disc(L) \leq \prod_i |v_i|$ is true for any (not necessarily LLL-reduced) basis and is known as Hadamard's inequality. The proof of the proposition is simple.
The reduction of any basis to an LLL-reduced basis can be desribed by an algorithm whose running time is $O(n^6(log(C)^3)$, where C is a bound for all $|v_i|$. In practice, it is often seen to take even less time.
Further, if the Gram matrix of the inner products $< v_i, v_j >$ of a basis $\{v_i\}$ is integral, the algorithm can be given in such a way that all computations are done in $\mathbf{Z}$ itself (and not go to $\mathbf{Q}$ as may be the case for a general basis).
The LLL algorithm does not give the shortest vector (this is a notoriously difficult problem) but one reasonably close to it.
One can adopt the LLL-algorithm to compute the kernel and image of an integral matrix also but the algorithm has to be modified to deal with dependent vectors also.

§ 3. **Factorisation of polynomials**

3.1 *GCD's of polynomials*

Multiplication of two polynomials of degrees m, n respectively, can be done by computing the values at $m+n+1$ points and using Lagrange's interpolation. The extended Euclidean algorithm to find the GCD of two polynomials works very well over finite fields. For example, for small primes p, and polynomials of degree of the order of 1000 over $\mathbf{F}_p$, the GCD is found in a few seconds. But, it can take much longer for polynomials over $\mathbf{Q}$. However, one can use a modification for integral polynomials where one works over $\mathbf{Z}$ itself (that is not divide polynomials really) and then it is much faster. It is referred to sometimes as pseudo-division, and can also be used to find GCD of integral poynomials. However, even that involves finding the content of the remainder at each step and there is a better algorithm which avoids that. This is the so-called subresultant algorithm which is similar to finding the resultant of two polynomials. The basic method of pseudo-division leading to a computation of the GCD of integral polynomials goes as follows.

Let $f, g \in \mathbf{Z}[X]$ be nonzero polynomials of degrees m, n (with $m \geq n$ say). Denote by $l(g)$ and $c(g)$ respectively, the leading coefficient and the content of g.

We want to find integral polynomials q, r such that $deg(r) < deg(g)$ and $l(g)^{m-n+1} f = qg + r$ where we want to avoid dividing polynomials.

Consider

$$q_1 = l(f)X^{m-n} \ , \ r_1 = l(g)f - q_1 g.$$

Notice that $l(g)f = q_1 g + r_1$ and $deg(r_1) < deg(f)$.

If $deg(r_1) < deg(g)$, then we are through by multiplying the above by a power of $l(g)$.

If $deg(r_1) \geq deg(g)$, define

$$q_2 = l(g)q_1 + l(r_1)X^{deg(r_1)-deg(g)} \ , \ r_2 = l(g)r_1 - (q_2 - l(g)q_1)g.$$

Note that $l(g)r_1 = (q_2 - l(g)q_1)g + r_2$ and that $deg(r_2) < deg(r_1)$. Proceeding in this manner, since the degrees of the r_i's decrease at each stage, we will get r_i with $deg(r_i) < deg(g)$ for some $i \leq m - n + 1$.

Note that $l(g)^i = qg + r_i$ for some q. This completes the pseudo-division of f by g.

To get the GCD of f, g, proceed as follows.

Do the pseudo-division of g by $r/c(r)$ and get a remainder of degree $< deg(r)$. Proceeding in this manner, it is clear that we are led to a constant polynomial after finitely many steps, we are led to the GCD of f, g upto a constant multiple. Then, the GCD of f, g is the corresponding primitive polynomial multiplied by $GCD(c(f), c(g))$.
An extension of this algorithm can be given where one can recover $u, v \in \mathbf{Z}[X]$ such that $fu + gv = GCD(f, g)$.

3.2 *Factorisation in* $\mathbf{F}_p[X]$
Factorisation in $\mathbf{Z}[X]$ or in $\mathbf{Q}[X]$ usually rely on factorisation in $\mathbf{F}_p[X]$ for various primes p. Unlike $\mathbf{Z}$, polynomials tend to have many factors over $\mathbf{F}_p$. One method is due to Berlekamp and uses the Chinese remainder theorem in $\mathbf{F}_p[X]$ and works well for square-free polynomials. We discuss a probabilistic algorithm due to Zassenhaus to factorise $f \in \mathbf{F}_p[X]$ completely.
Step I : Find square-free, relatively prime polynomials $f_1, \cdots, f_k \in \mathbf{F}_p[X]$ such that $f = f_1 f_2^2 f_3^3 \cdots f_k^k$.
To do this, one proceeds as follows.
The polynomial f_i would be the product of all linear factors $X - \alpha$ for various roots $\alpha \in \overline{\mathbf{F}_p}$ of f having multiplicity equal to i. These are found by taking GCD's with auxiliary polynomials as follows.
Note that $f = \prod_i f_i^i$ means $f' = \sum_i i f_i^{i-1} f_i' \prod_{j \neq i} f_j^j$ and that $g := (f, f') = \prod\{f_i^{i-1} : p \not| i\} \prod_{p|i} f_i^i$.
Start with $g_1 = g, h_1 = f/g = \prod\{f_i : p \not| i\}$.
Compute $h_{l+1} = h_l$ or (g_l, h_l) according as to whether $p|l$ or not.
Compute $g_{l+1} = \frac{g_l}{h_{l+1}}$.
Now, $h_l = \prod\{f_i : i \geq l, p \not| i\}$ by induction on l.
Similarly, $g_l = \prod_{p|i} f_i^i \prod\{f_j : j > l, p \not| j\}$ by induction.
Therefore, $f_l = \frac{h_l}{h_{l+1}}$ for all $p \not| l$.
When h_l is constant, then $g_{l-1} = \prod_{p|i} f_i^i = \alpha(X^p)$ for an easily computed α in $\mathbf{F}_p[X]$. Work with α in place of f all over again.
Step II : Split each f_i as $f_i = \prod_d f_{i,d}$ where $f_{i,d}$ is the product of *all* irreducible factors of degree d.
For this, note that the irreducible polynomials of degree d in $\mathbf{F}_p[X]$ are precisely the factors of $X^{p^d} - X$ which are *not* factors of $X^{p^e} - X$ for any $e < d$. Thus, one can take GCD's with various $X^{p^e} - X$ to find the irreducible factors of degree d and then $f_{i,d}$. To check irreducibility of a polynomial g

of degree d over $\mathbf{F}_p$, it is necessary and sufficient to check that $g/(X^{p^d} - X)$ and, for each prime $l|d$, $GCD(g, X^{p^{d/l}} - X) = 1$. It should be noted that in finding GCD's as above, one deals with powers like $X^{p^{d/l}}$ by looking at it mod g and then looking for GCD's etc.
(Main) Step III : Split each $f_{i,d}$ into the irreducible factors of degree d.
Renaming, let $g \in \mathbf{F}_p[X]$ be so that all its irreducible factors are of degree d. Let us take p odd (an analogue can be given for $p = 2$). Note that for any $h \in \mathbf{F}_p[X]$, one has

$$g = (g, h)(g, h^{(p^d-1)/2} + 1)(g, h^{(p^d-1)/2} - 1).$$

Therefore, we take a random monic polynomial h of degree $\leq 2d - 1$; then we will have (with probability $1/2$) a proper factor $(g, h^{(p^d-1)/2} - 1)$ of g.
Thus, we continue to work with smaller degree factors until ultimately we obtain all the irreducible factors of our g.
Finally, we can gather together all identical factors and order them by degree.

3.3 *Factorisation in* $\mathbf{Z}[X]$
As mentioned earlier, this very crucial aspect in algorithmic number theory will depend on factorisation over finite fields. We shall see soon that a polynomial time (in degree) algorithm exists for factorisation of primitive integral polynomials if we use LLL. However, it is interesting to note that no polynomial time algorithm is known for factorisation in $\mathbf{F}_p[X]$!
But, let us first make two simple observations.
First of all, an integral polynomial which is irreducible modulo p for *some* prime p is evidently irreducible over $\mathbf{Z}$ itself (apart from its content). But, it is a rare occurrence.
The second observation is more useful. Given any integral polynomial f of degree n, say, the degrees of the irreducible factors modulo any prime p, give a partition of n. If the partitions obtained modulo different primes are 'incompatible', once again f has to be irreducible. For example, if the partitions are $1 + 3$ and $2 + 2$ modulo two primes for a polynomial f of degree 4, f has to be irreducible over $\mathbf{Z}$.
However, in the absence of such obvious occurrences, in general, one tries to factorise a given f mod some prime p and lift the factorisation to a factorisation mod p^α for a suitable power and then lift it all the way to $\mathbf{Z}$.
To make such a thing work, one usually needs an a priori bound for the coefficients of *all* factors of f in $\mathbf{Z}[X]$. Such a bound is provided by the following

result of M.Mignotte :
If $g = \sum_{i=0}^{m} b_i X^i$ *divides* $f = \sum_{i=0}^{n} a_i X^i$ *in* $\mathbf{Z}[X]$, *then*

$$|b_i| \leq \binom{m-1}{i} (\sum_j |a_j^2|)^{1/2} + \binom{m-1}{i-1} |a_n|.$$

To see how such a bound is used, let us look at an example.
Example : Let $f = X^6 - 6X^4 - 2X^3 - 7X^2 + 6X + 1$.
For any factor g of degree ≤ 3, and any $i \leq 3$, $|b_i| \leq 23$ by the Mignotte bound. Consider a prime $p > 46$, say 47 mod which f is square-free.
Indeed, f mod 47 is $(X - 22)(X - 13)(X - 12)(X + 12)(X^2 - 12X - 4)$.
Firstly, as $f(0) = 1$, f has no roots in $\mathbf{Z}$ (as ± 1 are not roots mod 47).
To check that f has no quadratic factors, we look at the 2-term products like 22×13 etc., none of which are ± 1 mod 47.
Thus, f must either be irreducible or a product of two irreducible polynomials of degree 3. Suppose the latter happens. Then, one of the factors of degree 3 has $(X^2 - 12X - 4)$ as a factor mod 47. Since the constant term is ± 1, and $(-4)(\pm 12) \equiv \pm 1$ mod 47, one of the cubic factors of f reduces mod 47 to either $(X^2 - 12X - 4)(X - 12)$ or to $(X^2 - 12X - 4)(X + 12)$.
The first case is ruled out because the integral polynomial reducing to $X^3 + 23X^2 - X + 1$ mod 47 contradicts Mignotte's bound 13 for the coefficients of any possible cubic factor.
The second case $X^3 - 7X - 1$ is indeed possible, and in fact, leads to the factorisation

$$X^6 - 6X^4 - 2X^3 - 7X^2 + 6X + 1 = (X^3 + X - 1)(X^3 - 7X - 1)$$

in $\mathbf{Z}[X]$.

In general, here is how factorisation over $\mathbf{Z}[X]$ is carried out.
Start with $0 \neq f \in \mathbf{Z}[X]$.
Reduce to a square-free primitive polynomial (by $f \mapsto \frac{f}{(f,f')}$, $f \mapsto f/c(f)$).
Find p for which f is square-free mod p; that is, such that $GCD(f, f') = 1$ in $\mathbf{F}_p[X]$.
Find factorisation of f mod p.
Use Mignotte to find a bound B for the coefficients of any potential factor in $\mathbf{Z}[X]$ of degree $\leq deg(f)/2$.

Find the smallest α such that $p^\alpha > 2Bl(f)$, where $l(f)$ is the leading coefficient of f.
Lift factorisation of f mod p to a factorisation mod p^α using Hensel's lemma. Therefore, $f \equiv l(f)f_1 \cdots f_r \bmod p^\alpha$ with f_i monic polynomials in $\mathbf{Z}[X]$.
Starting with $d = 1$, consider each combination of products $f_{i1} \cdots f_{id}$, and find the unique $g \in \mathbf{Z}[X]$ whose coefficients have absolute value $< p^\alpha/2$, and which satisfies $g = l(f)f_{i1} \cdots f_{id}$ if $deg(g) \leq deg(f)/2$ and $g = l(f)\frac{f}{\prod_{l=1}^{d} f_{il}}$ if $deg(g) > deg(f)/2$.
If $g | l(f)f$ in $\mathbf{Z}[X]$, output the factor $F = g/c(g)$ and the power to which g divides f.
Take the new f to be f/F and drop all the corresponding f_i's in the list of factors mod p^α. Work with these now.
Go to $d = 2$ and repeat as above. Keep increasing d until it exceeds $r/2$ (remember r is the number of factors of f mod p^α).
If $d > r/2$, terminate and output $f/c(f)$.
This algorithm as written takes *exponential time*. But, one can get a polynomial-time algorithm by using LLL-reduction to certain lattices and this method leads to a factorisation of a polynomial in $\mathbf{Q}[X]$ also.
Let us see how LLL comes into the picture.

The basic result on which the algorithm is based is the following :
Let p be a prime, $k \in \mathbf{N}$, $f \in \mathbf{Z}[X]$ of degree $n > 0$, $h \in \mathbf{Z}[X]$ monic such that h mod p is irreducible, h mod p^k divides f mod p^k and (h mod p)^2 does not divide f mod p. Then, there is an irreducible factor $h_0 \in \mathbf{Z}[X]$ of f determined uniquely upto sign such that h mod p divides h_0 mod p. Furthermore, a factor g of f in $\mathbf{Z}[X]$ is divisible by h_0 in $\mathbf{Z}[X]$ if, and only if, g mod p^k is divisible by h mod p^k. In particular, h_0 mod p^k is divisible by h mod p^k.

We would like to find a way to compute h_0 efficiently. To do this, one (starts with f, p, k, h as above and) fixes some $m \geq l := deg(h)$ and considers the lattice L consisting of all integral polynomials of degree $\leq m$ which are, mod p^k, divisible by h mod p^k. This is a lattice in the vector space $\mathbf{R} + \mathbf{R}X + \cdots + \mathbf{R}X^m$ which we can think of as $\mathbf{R}^{m+1}$. Note that the Euclidean length provides the notion of the length of a polynomial. That is, $|\sum_{i=0}^{m} a_i X^i| = (\sum |a_i|^2)^{1/2}$.
Observe that L has a basis $\{p^k, p^k X, \cdots, p^k X^{l-1}, h, Xh, \cdots, X^{m-l}h\}$.

Note that h_0 itself belongs to L if, and only if, $deg(h_0) \leq m$. Now, it can be checked that an element $b \in L$ satisfying the condition $|b|^n < \frac{p^{kl}}{|f|^m}$ is divisible by h_0 in $\mathbf{Z}[X]$. (This requires proof which is not difficult, and see the LLL-paper).
In paticular, such an element b gives a factor $GCD(b, f)$ of f of degree > 1.
Let us now choose $\{b_1, \cdots, b_{m+1}\}$ to be an LLL-reduced basis. If one had the condition $p^{kl} > 2^{mn/2}\binom{2m}{m}^{n/2}|f|^{m+n}$, then one may use Mignotte's result quoted earlier to prove :
$b_1| < (p^{kl}/|f|^m)^{1/n}$ *if, and only if,* $deg(h_0) \leq m$.
The outline of the proof of this claim goes as follows.
Assume that $deg(h_0) \leq m$. Then, $h_0 \in L$. Then, the statement (d) of the proposition on LLL-reduced bases tells us that $|b_1| \leq 2^{m/2}|h_0|$.
Now, Mignotte's bound gives

$$|h_0| \leq \binom{2m}{m}^{1/2}|f|.$$

Thus, $|b_1| \leq 2^{m/2}\binom{2m}{m}^{1/2}|f|$.
Thus, if one had the condition $p^{kl} > 2^{mn/2}\binom{2m}{m}^{n/2}|f|^{m+n}$, then $|b_1|^n|f|^m < p^{kl}$.
This proves the claim and thus, the idea would be to get m etc. so that the above inequality is satisfied and ensure that $h_0 \in L$.
Moreover, if $t < m + 1$ is the largest for which b_t satisfies the inequality $|b_t|^n|f|^m < p^{kl}$, then it can be shown that all the b_j with $j \leq t$ also satisfy it and, $h_0 = GCD(b_1 \cdots, b_t)$ and $deg(h_0) = m + 1 - t$.
Therefore, in order to describe h_0 through an algorithm, one starts with f, p and a monic integral h which is irreducible mod p, divides f mod p and its square does not divide f mod p.
We may assume $l < n$; otherwise f is irreducible.
Find the smallest k satisfying $p^{kl} > 2^{n(n-1)/2}\binom{2(n-1)}{(n-1)}^{n/2}|f|^{2n-1}$.
We may use Hensel to modify h and assume that h divides f mod p^k also. Assume also that the coefficients of h are reduced mod p^k.
Now, if u is the largest natural number for which $l \leq \frac{n-1}{2^u}$, then we start with $m = [\frac{n-1}{2^u}]$ and consider the lattice L that we described above.
We choose an LLL-reduced basis $\{b_i\}$ as before.
Then, as asserted above, either $|b_1| \geq (p^{kl}/|f|^m)^{1/n}$ (in which case $\deg(h_0) >$

m), or $|b_1| < (p^{kl}/|f|^m)^{1/n}$ (in which case $\deg(h_0) \leq m$, $h_0 = GCD(b_1, \cdots, b_t)$ for some $t < m+1$).
In the latter case, we have determined h_0. In the former case, we change m to $[\frac{n-1}{2^{u-1}}]$ and repeat the argument. If $deg(h_0) > [\frac{n-1}{2^{u-1}}]$, change m to $[\frac{n-1}{2^{u-2}}]$ etc. and continue until $n-1$. If $\deg(h_0) > n-1$, then obviously $h_0 = f$ and we stop.

3.4 *Factorisation in* $K[X]$
Let K be an algebraic number field and $\mathcal{O}_K$, its ring of integers. We make some comments about how the algorithm of the previous section carries over to polynomials in $K[X]$. Also, a polynomial $f \in K[X]$ can be multiplied by an element of K^* to get a polynomial in $\mathcal{O}_K[X]$. However, it does not make sense to factorise over $\mathcal{O}_K$ as it is not usually a PID (=UFD). However, it is easy to use the factorisation over $\mathbf{Q}$ to get one over K as follows. If $\sigma_1, \cdots, \sigma_n$ are the various embeddings of K in $\mathbf{C}$ extending the inclusion $\mathbf{Q} \subset \mathbf{C}$, then the 'norm polynomial' $N(f) \in \mathbf{Q}[X]$ can be factorised as above into irreducibles. If f is square-free, then $N(f)$ is also square-free (over $\mathbf{Q}$) and it is easy to see that the factorisation $N(f) = \prod_{i=1}^{r} f_i$ into irreducibles over $\mathbf{Q}$ gives the factorisation $f = \prod_{i=1}^{r} GCD(f, f_i)$ into irreducibles over K. Here, of course, the GCD is in $K[X]$.
However, one could directly imitate the method for $\mathbf{Q}$, taking f in $\mathcal{O}_K[X]$, factorising it modulo a suitable prime ideal P and lifting to an appropriate power of it (a generalisation of Mignotte's bound is available) and use the LLL-reduced bases for powers of P (viewed as lattices) etc., and have a polynomial-time algorithm. Care is required in the choice of the power of the prime ideal above because that is crucial in ensuring a lift in $\mathcal{O}_K[X]$.
Note that an irreducible polynomial in $\mathcal{O}_K[X]$ may be reducible in $K[X]$.

§ 4. **Computing unit group and class group**

4.1 *Computing r_1, r_2 for K*
Given a number field $K = \mathbf{Q}(\theta)$, with $\theta \in \mathcal{O}_K$ and the minimal polynomial $f = min(\theta, \mathbf{Z})$, it is possible to obtain the numbers r_1, r_2 of real and non-real places without actually finding all the real roots. This is a method due to Sturm using sign changes, similar to the Descartes rule of signs. Briefly, it goes as follows :
Start with $F = f/c(f)(= f), G = f'/c(f'), a = 1, b = 1, s = sign(l(F))(= 1), n = deg(F), t = (-1)^{n-1}s, r_1 = 1$.
Perform the GCD operation over $\mathbf{Z}$; that is, put $d = deg(F) - deg(G)$ and compute polynomial remainder R with $l(G)^{d+1}F = QG + R$.
If $l(G) > 0$ or if d is odd, change R to $-R$.
If $sign(l(R)) \neq s$, change s to $-s$, r_1 to $r_1 - 1$.
If $sign(l(R)) \neq (-1)^{deg(R)}t$, change t to $-t$ and r_1 to $r_1 + 1$.
If $deg(R) = 0$, stop and output the value of r_1 (and $r_2 = (n - r_1)/2$).
If not, change F to G, G to $R/ab^d, a$ to $|l(F)|, b$ to $b^{1-d}a^d$ and go back to do division of this new F by G.

4.2 *Standard representation*
How does one represent elements of a number field K or of its ring of integers etc.? As has been hinted earlier, we want to view them as subsets of Euclidean spaces and be in a position to apply algorithms like LLL.
If $K = \mathbf{Q}(\theta)$ has degree n, where $\theta \in \mathcal{O}_K$, then every $\alpha \in K$ has a unique representation $\alpha = \frac{1}{d}\sum_{i=0}^{n-1} a_i\theta^i$ with $a_i \in \mathbf{Z}, d \in \mathbf{N}$ and $GCD(a_0, \cdots, a_{n-1}, d) = 1$.
Then, the standard representation of α is as $(a_0, a_1, \cdots, a_{n-1}, d) \in \mathbf{Z}^{n+1}$.
It is easy to use the standard representation of α to compute the characteristic polynomial χ_α of R_α (the 'multiplication by α map on the $\mathbf{Q}$-vector space K). Indeed, if we take $f = min(\theta, \mathbf{Z})$ and $F(X) = \sum_{i=0}^{n-1} a_i X^i$, then

$$\chi_\alpha(X) = d^{-n} R_Y(f(Y), dX - F(Y))$$

where R_Y denotes the resultant of polynomials in Y.
This expression for $\chi_\alpha(X)$ holds because

$$\chi_\alpha(X) = \prod_i (X - \sigma_i(\alpha)) = \prod_i (X - \frac{1}{d}F(\sigma_i(\theta))) = d^{-n}\prod_i (dX - F(\theta_i)).$$

4.3 *Checking isomorphism of fields*
To check whether $\mathbf{Q}(\alpha)$ is isomorphic to a subfield of $\mathbf{Q}(\beta)$, there are a number of ways. We want to check if some conjugate of α is in $\mathbf{Q}(\beta)$. One way is to use factorisation. Decomposing $f = min(\alpha, \mathbf{Q})$ in $\mathbf{Q}(\beta)[X]$ as a product of irreducibles, one would have a linear factor if, and only if, some conjugate of α is in $\mathbf{Q}(\beta)$. To check isomorphism of fields, one simply checks their degrees and checks if one is isomorphic to a subfield of the other.

4.4 $\mathcal{O}_K$ *and ideals as lattices*
Let $K = \oplus_{i=1}^{n} \mathbf{Q}\alpha_i$ be a number field and let $R = \oplus_{i=1}^{n} \mathbf{Z}\alpha_i$. This is an abelian subgroup of K having rank n. Then, consider any abelian subgroup M of K of rank n, one can define the denominator of M with respect to R to be the smallest natural number $d = d(M)$ such that $dM \subseteq R$.
The *HNF-basis of* M *with respect to* R is defined to be the unique basis $\{\omega_1, \cdots, \omega_n\}$ of M such that $d\omega_j = \sum_i \omega_{ij}\alpha_i$ and the triangular integral matrix (ω_{ij}) is in HNF.
In the special case of an order $R = \mathbf{Z}[\theta]$ where $K = \mathbf{Q}(\theta)$, the HNF with respect to R is particularly nice; it is $\omega_j = \frac{z_j}{d}(\theta^{j-1} + \sum_{i<j} h_{ij}\theta^{i-1})$ with z_i natural numbers such that $z_j | z_i$ for $j > i$ and $0 \leq h_{ij} < \frac{z_i}{z_j}$ for $i < j$.
Actually, z_1 is the smallest positive element of $dM \cap \mathbf{Z}$.
Note that as an advantage of using such an HNF-basis, one can compute the norm of any ideal I of $\mathcal{O}_K$ as follows.
First, let $K = \mathbf{Q}(\theta)$ where $\theta \in \mathcal{O}_K$ and let $R = \mathbf{Z}[\theta]$ as above. Of course, $R \subseteq \mathcal{O}_K$ and equality may not hold.
For an ideal I of $\mathcal{O}_K$, consider the HNF matrix (h_{ij}) of I with respect to R and the denominator d of I with respect to R. Then,

$$d^{-n}N(I) = [\mathcal{O}_K : dI] = [\mathcal{O}_K : R]\prod_i h_{ii}.$$

Multiplication of ideals is again carried out efficiently by using HNF-bases as follows. Let I, I' be integral ideals given by HNF-matrices M, M' with respect to some integral basis $\{\omega_1, \cdots, \omega_n\}$. That is, the columns of M give co-ordinates in terms of the integral basis above, for a basis of I etc. If γ_i and γ'_j are the various column vectors of M, M' respectively, then one looks at the $n \times n^2$ matrix whose columns are the co-ordinates of $\gamma_i\gamma'_j$ in terms of the integral basis. Compute the HNF of this $n \times n^2$ matrix, and this is the HNF for II'.

In practice, one can be even more efficient by getting hold of a 2-element generating set for one of the ideals, say I', and working with an $n \times 2n$ matrix instead of an $n \times n^2$ matrix.
For operating with the class group, one needs to compute the inverse of any ideal class, and this requires the computation of the *different*.
Recall that the different $\delta(K)$ is the integral ideal whose inverse is the fractional ideal

$$\{x \in K : Tr_{K/\mathbf{Q}}(x\mathcal{O}_K) \subseteq \mathbf{Z}\}.$$

To find it, start with an integral basis $\{\omega_1, \cdots, \omega_n\}$ and compute the inverse of the matrix $T = (Tr_{K/\mathbf{Q}}(\omega_i\omega_j))$. Then, the columns of this inverse matrix when considered as co-ordinates on the basis $\{\omega_1, \cdots, \omega_n\}$ give a $\mathbf{Z}$-basis of the fractional ideal $\delta(K)^{-1}$.
Further, if I is an integral ideal given by a matrix $M = (m_{ij})$ whose columns give co-ordinates for a $\mathbf{Z}$-basis of I, then the columns of $({}^tMT)^{-1}$ give a basis for $I^{-1}\delta(K)^{-1}$. Thus, I^{-1} is found by looking at the above method to compute the basis for the inverse $(I\delta(K)^{-1})^{-1}$ but since $I\delta(K)^{-1}$ may not be an integral ideal, one usually modifies the method as follows. Firstly, we note that the matrix T above has determinant $disc(K)$, the discriminant of K and therefore $disc(K)\delta(K)^{-1}$ is an integral ideal.
We are given the integral ideal I as a matrix $M = (m_{ij})$ whose columns give the co-ordinates of a $\mathbf{Z}$-basis $\{\gamma_j\}$ of I in terms of a given integral basis $\{\omega_1, \cdots, \omega_n\}$. We wish to compute the HNF-basis of I^{-1}. Let us proceed as follows.
Compute $T, det(T)$ and $det(T)T^{-1}$.
Call $\{\delta_j\}$, the columns of $det(T)T^{-1}$; they give a basis for the integral ideal $disc(K)\delta(K)^{-1}$. Compute N, the HNF of the $n \times n^2$ matrix whose columns are the co-ordinates on the n^2 products $\gamma_i\delta_j$; evidently the columns of N give a basis for $disc(K)I\delta(K)^{-1}$.
Put $P = disc(K)({}^tNT)^{-1}$, and let d be the common denominator for entries of P. Compute W, the HNF of dP. Then, W is the HNF of I^{-1} and d is the denominator of I^{-1} (with respect to $\mathcal{O}_K$).

4.5 *Computing an integral basis for* $\mathcal{O}_K$
In each computation above, we pre-supposed that we know $\mathcal{O}_K$ and that we, in fact, know an integral basis. However, it is a highly nontrivial problem to find $\mathcal{O}_K$. As a matter of fact, finding $\mathcal{O}_K$ is equivalent to finding the square-free part of a given natural number. As of now, one does not know

if this latter problem is any easier than the hard problem of factorisation of natural numbers.
If $K = \mathbf{Q}(\theta)$ with $\theta \in \mathcal{O}_K$ of degree n, the discriminant of the number field K satisfies $disc(1, \theta, \cdots, \theta^{n-1}) = disc(K)[\mathcal{O}_K : \mathbf{Z}[\theta]]^2$.
Therefore, a criterion for $\mathcal{O}_K$ to be equal to $\mathbf{Z}[\theta]$ is that $disc(1, \theta, \cdots, \theta^{n-1})$ is square-free.
Due to this difficulty, one sometimes tries to work with orders like $\mathbf{Z}[\theta]$ in place of the maximal order $\mathcal{O}_K$ but this raises other complications. For instance, not all ideals over an order may be invertible; of course, all invertible ideals do form a group called the class group of that order.
A method to obtain $\mathcal{O}_K$ starting from an order like $\mathbf{Z}[\theta]$ is to deal with the primes dividing the index $[\mathcal{O}_K : \mathbf{Z}[\theta]]$ one at a time and enlarging the order to get to the maximal order. This works by applying the famous result of Dedekind on the reciprocity for prime ramification and is due to Zassenhaus; it goes as follows.

First, write $K = \mathbf{Q}(\theta)$ with $\theta \in \mathcal{O}_K$, $f = min(\theta, \mathbf{Q})$ of degree n. Write $disc(f) = dc^2$ where d is a fundamental discriminant (or 1). Hence, the only possible primes dividing the index $[\mathcal{O}_K : \mathbf{Z}[\theta]]$ are among the prime divisors of c.
Start with the order $A = \mathbf{Z}[\theta]$ and any prime $p|c$.
We shall first use the following steps to check whether A is *p-maximal*; that is, whether p does not divide the index $[\mathcal{O}_K : \mathbf{Z}[\theta]]$.
Write $f \bmod p$ (say $\bar{f}$) as the product $\prod_{i=1}^{g} \bar{f}_i^{e_i}$ of monic irreducibles in $\mathbf{F}_p[X]$.
Let $f_i \in \mathbf{Z}[X]$ be arbitrary monic lifts of $\bar{f}_i$.
Let $h \in \mathbf{Z}[X]$ be a monic lift of $\prod_{i=1}^{g} \bar{f}_i^{e_i - 1}$.
Look at the polynomial $g(X) = \frac{1}{p}(h(X) \prod_{i=1}^{g} f_i(X) - f(X)) \in \mathbf{Z}[X]$.
Then, $\mathbf{Z}[\theta]$ is p-maximal $\Leftrightarrow GCD(\bar{g}, \bar{h}, \prod_{i=1}^{g} \bar{f}_i) = 1$ in $\mathbf{F}_p[X]$.
Now, if $A := \mathbf{Z}[\theta]$ is not p-maximal, then for any monic lift $u(X) \in \mathbf{Z}[X]$ of $\frac{\bar{g}}{GCD(\bar{g}, \bar{h}, \prod_{i=1}^{g} \bar{f}_i)}$, one has the enlarged order $A' = \mathbf{Z}[\theta] + \frac{1}{p} u(\theta)\mathbf{Z}[\theta]$ which satisfies $[A' : A] = p^m$ where m is the degree of $GCD(\bar{g}, \bar{h}, \prod_{i=1}^{g} \bar{f}_i)$.
Thus, note that $disc(A') = \frac{1}{p^{2m}} disc(f)$.
So, if $p^2 \not| disc(A')$, then A' is p-maximal, and one goes to the next prime divisor of c.
If $p^2 | disc(A')$, then work with A' in place of A and continue as above.

4.6 *Decomposing prime ideals in* $\mathcal{O}_K$

If $K = \mathbf{Q}(\theta)$ with $\theta \in \mathcal{O}_K$, then we know the decomposition of any unramified integral prime p which does not divide $[\mathcal{O}_K : \mathbf{Z}[\theta]]$ from the decomposition mod p of the minimal polynomial of θ over K. This is Kummer's theorem. So, let us look at a prime p dividing this index; that is, a prime p for which $\mathbf{Z}[\theta]$ is not p-maximal. More generally, instead of the whole of $\mathcal{O}_K$, the following procedure due to Buchmann and Lenstra works for any order $\mathcal{O}$ which contains $\mathbf{Z}[\theta]$ for which $p|[\mathcal{O} : \mathbf{Z}[\theta]]$.
Let $p\mathcal{O} = \prod_{i=1}^{g} P_i^{e_i}$.
Consider the ideal $I_p = \prod_{i=1}^{g} P_i$. The point is that the $\mathbf{F}_p$-algebras $\mathcal{O}/P_i^{e_i}$ may not be separable, and one wants to get to separable algebras over the finite field and lift the information. In order to do this, we define

$$K_j = I_p^j + p\mathcal{O} = \prod_{i=1}^{g} P_i^{min(e_i,j)}.$$

Then, $K_j \subseteq K_{j-1}$, and so $J_j := K_j K_{j-1}^{-1} = \prod_{e_i \geq j} P_i$.
Note $J_j \subseteq J_{j+1}$. So, $H_j := J_j J_{j+1}^{-1} = \prod_{e_i=j} P_i$ is a product of distinct maximal ideals.
Moreover, H_j's are pairwise coprime, and $p\mathcal{O} = \prod_{j=1}^{max(e_1,\ldots,e_g)} H_j^j$.
(This is exactly the anlogue of the square-free factorisation in $\mathbf{F}_p[X]$ we discussed earlier.)
Thus, we need to say how to compute the H_j's. Now, $\mathcal{O}/H_j = \mathbf{F}_p[\bar{\alpha_j}]$, as it is a separable $\mathbf{F}_p$-algebra.
Let $\bar{h_j}$ be the characteristic polynomial of $\bar{\alpha_j}$ over $\mathbf{F}_p$.
For any lift $h_j \in \mathbf{Z}[X]$, let $h_j \equiv \prod_{i=1}^{t_j} u_{ij} \bmod p$ where $\bar{u_{ij}}$ are irreducibles.
Then, the ideals $U_{ij} := H_j + u_{ij}(\alpha_j)\mathcal{O}$ are maximal, and $H_j = \prod_{i=1}^{t_j} U_{ij}$.
It should be remarked that the finding a primitive element of a separable $\mathbf{F}_p$-algebra (and therefore splitting H_j into the product of maximal ideals) is not as time-consuming as multiplying and dividing ideals ! Therefore, to speed up these latter jobs, one observes that our multiplication of ideals etc. takes place only mod $p\mathcal{O}$. It is quicker to multiply or divide mod $p\mathcal{O}$ using easy linear algebra over $\mathbf{F}_p$.

4.7 *Roots of unity in K*
We start first by recalling some basic objects. Recall the 'log' map that one introduces while proving Dirichlet's unit theroem for a number field K.
If $\sigma_1, \cdots, \sigma_{r_1}$ are the real embeddings of K and $\sigma_{r_1+1}, \cdots, \sigma_{r_1+r_2}, \bar{\sigma_{r_1+1}}, \cdots, \bar{\sigma_{r_1+r_2}}$ are the complex (that is, non-real) embeddings, the 'log' map is the group

homomorphism

$$`log' : K^* \to \mathbf{R}^{r_1+r_2}$$

$$x \mapsto (log|\sigma_1(x)|, \cdots, log|\sigma_{r_1}(x)|, 2log|\sigma_{r_1+1}(x)|, \cdots, 2log|\sigma_{r_1+r_2}(x)|).$$

$Image(log)$ is contained in the hyperplane $\{v : \sum_i v_i = 0\}$, and $Ker(log) = \mu(K)$. Further, $log(\mathcal{O}_K^*)$ is a lattice in $Image(log)$ and its volume is called the regulator of K and is denoted by $reg(K)$.
Of course, $\mu(K)$ is just the kernel of the 'log' map above. Being a finite, cyclic group it suffices thus to find $|\mu(K)|$.
One uses LLL-reduction to do this. Given a real, symmetric $n \times n$ positive-definite matrix A and some $c > 0$, LLL-reduction can be used to find *all* nonzero vectors $x \in \mathbf{Z}^n$ such that $q_A(x) \leq c$, and finds also all the corresponding values $q_A(x)$. One first uses Gram-Schmidt or Cholesky to find R such that $A =^t RR$ and then applies LLL-reduction to the row vectors of R^{-1}. To adopt this for computing $|\mu(K)|$, we follow a method of Fincke and Pohst, as follows.
Assume that we 'know' θ, its conjugates, and an integral basis $\{\omega_i\}_{i=1}^n$of K (as polynomials in θ).
We are looking for elements of $Ker(log)$; that is, for integers $x_1, \cdots, x_n$ such that all the conjugates of $\sum_{i=1}^n x_i\omega_i$ are on the unit circle.
Thus, we want $q(x) := \sum_j |\sigma_j(\sum_i x_i\omega_i)|^2 = n$.
Note that for arbitrary integers $y_1, \cdots, y_n$, the AM-GM inequality gives

$$\sum_j |\sigma_j(\sum_i y_i\omega_i)|^2 \geq n(\prod_j |\sigma_j(\sum_i y_i\omega_i)|)^{2/n} \geq n$$

with equality if and only if $|\sigma_j(\sum_i x_i\omega_i)|^2$ are equal for all j.
In other words, the minimum value of the above quadratic form on $\mathbf{Z}^n$ is n, and it is attained at $(y_1, \cdots, y_n)$ if and only if $\sum_{i=1}^n y_i\omega_i \in \mu(K)$.
Start with the matrix $A = (a_{ij})$, where $a_{ij} = \sum_k \sigma_k(\omega_i)\sigma_k(\omega_i)$.
Of course, one takes 'good approximations to a_{ij} just as one takes 'good' approximations to θ.
Then, with $c = n + 0.1$, one obtains all points $x \in \mathbf{Z}^n$ with $q(x) \leq c$. Then, one can check each corresponding $\sum_i x_i\omega_i$ to see if it is a root of unity. The actual order of $\mu(K)$ will be twice the number so obtained.

4.8 *Approach to computing the class group*
First of all, we remark that one application of class group computation is to

factorisation of natural numbers. Indeed, factorising n can be proved to be equivalent to the computation of the 2-Sylow subgroup of the class group of $\mathbf{Q}(\sqrt{-n})$.

Computations of the class number, the regulator and a fundamental system of units are all inter-related. Note that the class number is not any reasonable function of the discriminant of the field; that is, discriminants very close to each other can correspond to very different class numbers. Similarly, the regulator is also not a good function of the discriminant. However, the nice thing is that the product of the class number and the regulator varies well with the discriminant of the field as evinced by Dedekind's formula for the residue of the Dedekind zeta function at 1. *This is what is exploited in forming algorithms for the class number and the regulator.*

Recall that the regulator $reg(K)$ is defined using any fudamental system of units (a basis of the free abelian part of the unit group) $u_1, \cdots, u_{r_1+r_2-1}$ as the absolute value of the determinant of ay $(r_1 + r_2 - 1) \times (r_1 + r_2 - 1)$ submatrix of the $(r_1 + r_2 - 1) \times (r_1 + r_2)$ matrix $(log||\sigma_j(u_i)||)_{ij}$ where $||x||$ stands for $|x|$ or $|x|^2$ according as to whether x is real or not.

Here are three results which we shall need :

(I) (Minkowski) Each ideal class has an ideal of norm $\leq (4/\pi)^{r_2} \frac{n!}{n^n} \sqrt{|disc(K)|}$.

(II) (Dedekind's formula for residue of $\zeta_K(s)$ at $s = 1$:)
$\frac{h(K)reg(K)2^{r_1}(2\pi)^{r_2}}{|\mu(K)|\sqrt{|disc(K)|}} = \prod_p \{(1 - \frac{1}{p}) \prod_{P|p} (1 - \frac{1}{NP})^{-1}\}$.

(III) (assuming GRH) Non-inert prime ideals of norm $\leq 12(log|disc(K)|)^2$ generate the class group.

A brief, rough description of our procedure

The computation of $Cl(K)$ needs prior computation of $reg(K)$ and of $\mathcal{O}_K^*$. But, let us view it backwards. Suppose we have found fractional ideals $I_1, \cdots, I_r$ such that the classes $\bar{I}_1, \cdots, \bar{I}_r$ generate $Cl(K)$. Then, we have a surjection

$$\mathbf{Z}^r \to Cl(K) \ ; \ x \mapsto \prod_i \bar{I}_i^{x_i}$$

whose kernel is a lattice Λ (called the lattice of relations) which we want to find. For the moment, suppose we have found some number m of relations sufficient to generate Λ.

Put these relations as column vectors of an $r \times m$ matrix M (of course, $\Lambda = Image(M : \mathbf{Z}^m \to \mathbf{Z}^r)$).

We first find a *basis* of Λ in terms of the m generators, using HNF. That is,

we get some $U \in GL(m, \mathbf{Z})$ such that $MU = (0\ H)$ where 0 is $r \times (m-r)$ and H is $r \times r$ and is in HNF.
Then, the columns of H form a basis for Λ.
In order to determine the class group, apply the elementary divisor theorem (that is, the SNF) to get $H = V\ diag(d_1, \cdots, d_r)\ V'$ with $d_i | d_{i+1}$.
If V_i is the i-th column of V (considered as an element of $\mathbf{Z}^r$), then

$$\mathbf{Z}^r/\Lambda = \oplus_{i=1}^{r} (\mathbf{Z}/d_i\mathbf{Z})\bar{V_i}$$

where $\bar{V_i}$ is the image of V_i mod Λ.
Note that the above is an equality - not just an isomorphism.
To get $Cl(K)$ (which is isomorphic to $\mathbf{Z}^r/\Lambda$) as something concrete, we look at the isomorphism $\mathbf{Z}^r/\Lambda \cong Cl(K)$. Then, we get

$$Cl(K) = \bigoplus_{i=1}^{r} (\mathbf{Z}/d_i\mathbf{Z}) \overline{\prod_{j=1}^{r} I_j^{a_{ji}}}$$

where $V = (a_{ij})$.
This finishes the rough sketch of the procedure to compute $Cl(K)$.
Therefore, we are reduced to determining generators for $Cl(K)$ and finding enough relations. The latter step proceeds by finding some generators and checking whether they suffice - this requires us to be able to compute $reg(K)$ - and then proceed to find more relations if they do not suffice.

4.9 *Computing* $reg(K), h(K)$
At the moment, let us start with some generators $\{\bar{I_i}\}_1^r$ of $Cl(K)$. Suppose we already have a few relations written in terms of a *relation matrix* $M = (m_{ij})_{r \times m}$; in other words, the j-th column of M gives a relation $\prod_{i=1}^{r} I_i^{m_{ij}} = \alpha_j \mathcal{O}_K$. In order to keep track of relations among ideals and not merely among ideal classes, we wish to keep along with M also track of the α_j's. The way to do this is by considering the 'complex log' map $L_{\mathbf{C}}$ defined as follows :

$$x \mapsto (log\sigma_1(x), \cdots, log\sigma_{r_1}(x), 2log\sigma_{r_1+1}(x), \cdots, 2log\sigma_{r_1+r_2}(x))$$

where the logarithm is defined only upto addition of integral multiples of $2i\pi$.
Then, one keeps the $r \times m$ matrix M and the m vectors α_j together as an $(r + r_1 + r_2) \times m$ matrix $\begin{pmatrix} M \\ M_{\mathbf{C}} \end{pmatrix}$.
Note that the first r rows are integral and the last $r_1 + r_2$ are complex. The

HNF-algorithm gives $U \in GL(m, \mathbf{Z})$ such that $MU = (\,0 \quad H\,)$ with H in HNF.
So, $\begin{pmatrix} M \\ M_{\mathbf{C}} \end{pmatrix} U = \begin{pmatrix} 0 & H \\ Z_{\mathbf{C}} & H_{\mathbf{C}} \end{pmatrix}$ for some $Z_{\mathbf{C}}$ and $H_{\mathbf{C}}$.
We notice that linear combinations of relations are relations - multiplying ideals corresponds to addition of exponents and the mutliplication of α's corresponds to the addition under the log map.
Thus, both $\begin{pmatrix} 0 \\ Z_{\mathbf{C}} \end{pmatrix}$ and $\begin{pmatrix} H \\ H_{\mathbf{C}} \end{pmatrix}$ are matrices of relations among ideals. The former matrix has $m - r$ colums; for each $j \leq m - r$, its j-th column corresponds to a unit u_j.
Thius, we note that this method already finds some $(m - r)$ units. Suppose, we have found $r_1 + r_2 - 1$ units; that is, suppose $m - r \geq r_1 + r_2 - 1$.
In such a case, any $r_1 + r_2 - 1$ columns of the real part of $Z_{\mathbf{C}}$ gives the matrix of the usual 'log' map. Now, let us use these $r_1 + r_2 - 1$ units $\epsilon_1, \cdots, \epsilon_{r_1+r_2-1}$ and consider the $(r_1 + r_2) \times (r_1 + r_2)$ matrix where the last column is the vector $(1, \cdots, 1, 2, \cdots, 2)$ with the first r_1 entries 1, and the first $r_1 + r_2 - 1$ similar to those in the definition of the regulator (for defining the regulator, recall that the units are fundamental).
In other words, our matrix has, for $j \leq r_1+r_2-1$, the (i, j)-th entry $log|\sigma_i(\epsilon_j)|$ if $i \leq r_1$, and $2log|\sigma_i(\epsilon_j)|$ if $i > r_1$.
The determinant of this matrix is a multiple (possibly 0 (!)) of $reg(K)$. The *hope* is that one gets a *non-zero, small* multiple $r'(K)$ of $reg(K)$. The point is that one can then check if $r'(K) = reg(K)$ using the following procedure. If unequal, it means that the number of relations found is insufficient.
Checking if $r'(K) = reg(K)$ turns out equivalent to simultaneously checking if $h'(K) = h(K)$.
Here, we mean by $h'(K)$, the tentative class number computed for the lattice Λ' of relations found so far (about which we do not know whether they suffice). Now, the residue is given in terms of an infinite product over all primes. But, GRH implies by a result of E.Bach that the subproduct over only the primes $\leq 12(log|disc(K)|)^2$ changes the infinite product only by a factor $< \sqrt{2}$. In other words, assuming the GRH, one has some positive real a such that

$$\frac{a}{\sqrt{2}} < h(K)reg(K) < a\sqrt{2}.$$

a is nothing but the expression $\frac{|\mu(K)|\sqrt{|disc(K)|}}{2^{r_1}(2\pi)^{r_2}} \prod_{p \leq 12(log|disc(K)|)^2} \frac{1-\frac{1}{p}}{\prod_{P|p}(1-\frac{1}{NP})^{-1}\}}$.

Therefore, if $h'(K)r'(K) = mh(K)reg(K)$, then the inequality $h'(K)r'(K) < a\sqrt{2}$ can hold if, and only if, $m = 1$ (because $m \geq 2$ would imply $h(K)reg(K) < \frac{a\sqrt{2}}{m} \leq \frac{a}{\sqrt{2}}$, an impossibility).
Thus, the upshot is that *once we compute the above expression a, we need only check if* $h'(K)r'(K) < a\sqrt{2}$.
If so, then $h'(K) = h(K), r'(K) = reg(K)$. Therefore, the units we have form a fundamental system of units. Of course, we have assumed the validity of GRH.
Now, if $h'(K)r'(K) \geq a\sqrt{2}$, we go about finding more relations, as said earlier. One hopes (and this happens in practice) that after a finite number of steps, $h'(K)r'(K) < a\sqrt{2}$ is satisfied.

4.10 *Starting to find generators and relations for* $Cl(K)$
As we have seen now, the discussion depends on our being able to find some generators and relations for the class group. In the previous section, we looked at the bound $12(log|disc(K)|)^2$; it turns out that (under GRH) this bound is also such that the class group is generated by all prime ideals of norm bounded by it. Without appealing to GRH, there is the Minkowski bound $(\frac{4}{\pi})^{r_2}\frac{n!}{n^n}\sqrt{|disc(K)|}$. Of course, a bound like $12(log|disc(K)|)^2$, is efficient to find generators of $Cl(K)$ - simply take all *prime* ideals of norm at most this number. However, the number of relations among them seems to be computationally rather large. So, in practice, one considers only the prime ideals $\{P_i\}_{i=1}^{r}$ of norm $\leq 0.1(log|disc(K)|)^2$; they lead to much fewer relations. Though these prime ideals need not generate the whole class group, they almost always do. Once a (nice) bound B such as above, say, is fixed, the set $\mathcal{P}$ of prime ideals of norm $\leq B$ is found by starting with any integral prime $p \leq B$ and factorising $p\mathcal{O}_K$. One refers to a choice of $\mathcal{P}$ as a *factor base*. Choosing random exponents x_i (preferably small, with most of them zero), and considering the ideal $I = \prod_{P_i \in \mathcal{P}} P_i^{x_i}$. There is a way of using LLL-algorithm to *reduce ideals along various directions* which leads to another ideal *in the same class* but having 'small' norm in a sense. Thus, we would have $\alpha \in K^*$ such that $J = \alpha I$ has small norm and then $JI^{-1} = \alpha\mathcal{O}_K$ is a relation in $Cl(K)$. This relation is hopefully made from primes in $\mathcal{P}$ only. Thus, under the assumption of GRH, we would have an efficient algorithm to determine some generators and some relations in $Cl(K)$ to start with.

4.11 *Checking if an ideal is principal*

This essentially involves solving the discrete log problem in $Cl(K)$. Let us take a factor base $\mathcal{P}$ as before. Look at only the ideals $I = \prod_{P_i \in \mathcal{P}} P_i^{x_i}$ with $x_i \leq d_i$; recall that $Cl(K) = \bigoplus_{i=1}^{r} (\mathbf{Z}/d_i\mathbf{Z})\bar{I}_i$.
Let X be the column vector made of the x_i's. Then, as the columns of the HNF matrix H (obtained during the computation of $Cl(K)$) form a basis of the relation lattice Λ, we have :
I is principal if, and only if, $H^{-1}X$ has integer entries.
This is easy to check as H is upper triangular.
If I is not principal, then one can find the x_i's - they are the fractional parts of the entries of $H^{-1}X$ multiplied by the corresponding d_i's.

§ 5. **Galois groups**

5.1 *A 'polynomial time' determination of Galois groups*

If K is a number field, and $f \in K[X]$, one is seeking a computation of the Galois group of (the splitting field of) f over K. Now, if we mean determination of $Gal_K(f)$ to to mean that we have a multiplication table for this group, there may be no polynomial time (in the input size) algorithm.

However, we may do one of two things : (i) look for an algorithm which is polynomial time in input plus output, or (ii) look for a polynomial time algorithm in the usual sense but express the answer (the Galois group) in some other way.

The second problem is not solved satisfactorily as yet. The first problem, on the other hand, is easy to solve affirmatively using the usual construction of $Spl_K(f)$.

Indeed, given a natural number N, one can decide whether $Gal_K(f)$ has order $\leq N$, actually give in this case all the elements, the multiplication table and an embedding in the symmetric group - all in "polynomial in $(N + l)$ time" where l is the input data size.

To see this, as usual, let us take an irreducible factor $g \in K[X]$ of degree > 1 of f and consider the field $L = K[X]/(g)$. If $[L : K] > N$ the Galois group certainly has order $> N$, and we stop. If not, continue with L in place of K and build the splitting field of f over L (which is also $Spl_K(f)$), and check at each stage whether the degree over K has exceeded N or not.

If the algorithm has run with the conclusion that the order of $Gal_K(f)$ is $\leq N$, then one can easily determine all its elements as follows.

This is similar to checking whether two given fields are isomorphic. Evidently, writing $Spl_K(f) = K(\alpha)$, each root of $min(\alpha, K)$ corresponds to a K-automorphism of $Spl_K(f)$. Thus, one can find all the elements and their products in time which is polynomial in $O(Gal_K(f)) + l$. This also gives an embedding of the Galois group into S_n where $n = deg(f)$.

5.2 *Deciding if Galois group is abelian*

The algorithm of the previous section applied with $N = n$ certainly gives a decision as to whether the Galois group is abelian because of the following fact :

A transitive abelian subgroup subgroup of S_n has order n.

Applying this to each irreducible factor of f, one can decide whether $Gal_K(f)$

is abelian (which happens exactly when $Gal_K(g)$ is abelian for each irreducible factor g).
However, even in the abelian case, the Galois group is not determined in time which is polynomial in the input data size.
In the special case of $K = \mathbf{Q}$ with $f = \prod_{i=1}^{n}(X^2 - a_i)$ for distinct $a_i \in \mathbf{Q}$, an algorithm has been given, but no such algorithm has been written down for general K. H.W.Lenstra Jr. believes that it can probably be done under the GRH.

5.3 *Deciding solvability*
This is obviously a very interesting and important problem and has been solved by S.Landau and G.Miller.
As a first guess, one could try to imitate the abelian case and check whether there is a polynomial (in n) bound for transitive solvable subgroups of S_n. But, no such bound exists. Very fortunately, such a bound is proved by Palfy to exist when one restricts oneself to *primitive* such groups.
Let us recall that a subgroup of S_n is primitive if its stabilisers of points are maximal subgroups. For $Gal_K(f)$, this means that there are no proper intermediate fields between K and $K(\alpha)$ for any root α of f.
Now, to reduce the general case to the primitive case, one finds a chain

$$K = K_0 \subset K_1 \subset \cdots K_t = K(\alpha)$$

which cannot be refined. Observe that this is what we need because $Gal_K(f)$ is solvable if, and only if, the Galois group of K_{i+1} over K_i is solvable for each i.
How does one find such a chain ? This is the problem of finding maximal proper intermediate subfields which was done by Landau and Miller. We must note that finding *all* subfields is not a tractable problem as there are too many.
To find maximal intermediate subfields, split f into monic irreducible factors over $K(\alpha)$. Of course, one factor is $X - \alpha$. For any other irreducible factor g, one has an intermediate proper subfield L_g of $K(\alpha)$ as follows.
If $g = X - \beta$, then writing $\sigma\alpha = \beta$, take $L_g = K(\alpha)^\sigma$.
If $\deg(g) > 1$, take a root β in an extension of $K(\alpha)$ and take $L_g = K(\alpha) \cap K(\beta)$.
Checking that all intermediate maximal proper subfields are among these is a consequence of the following purely group-theoretic observation.

Let G be a finite group and $H < J \leq G$ be subgroups with H a maximal proper subgroup of J. Then, any $g \in J \setminus H$ is so that

$$< H, g >= J \quad if \quad gHg^{-1} = H,$$

$$< H, gHg^{-1} >= J \quad if \quad gHg^{-1} \neq H.$$

It ought to be noted that this algorithm only decides whether the Galois group is solvable but does not determine it even if f is irreducible (unlike the abelian case).

§ 6. **Miscellaneous topics**

6.1 *Determining number fields from the zeta function*
We already saw in the computation of the class number that analytic information in terms of the Dedekind zeta function of a number field was used. A natural question is whether the zeta function determines the number field. Unfortunately, this is far from true and, indeed, different number fields with the same zeta function can be constructed using simple finite group theory. Actually, this was imitated by T.Sunada to construct so-called isospectral manifolds which are not isomorphic - the phenomenon has been referred to informally as 'one cannot hear the shape of a drum'. However, if two number fields are solvable by radicals (that is, are Galois over $\mathbf{Q}$ and have solvable Galois groups), they are isomorphic if, and only if, their zeta functions are the same provided we avoid certain degrees as follows.
If K, L are number fields, and X, Y are the sets of Q-embeddings of K, L respectively, in $\bar{\mathbf{Q}}$, then the absolute Galois group $G := Gal(\bar{\mathbf{Q}}/\mathbf{Q})$ acts transitively by permutations on X and Y. By Galois theory, X and Y are isomorphic as G-sets if, and only if, the fields K, L are isomorphic.
Further, if one calls X and Y *linearly equivalent* if the permutation characters of G on X and Y are the same (that is, each $g \in G$ fixes the same number of elements on X and Y), then such a thing happens if, and only if, K, L have the same Dedekind zeta function.
The following purely group-theoretic statement proved by De Smit and H.W.Lenstra Jr. ([3]) shows that a number field that is solvable by radicals and has degree n different from (ii) in the theorem, is determined upto isomorphism by its zeta function.

Theorem ([3]**)**
For any $n \in \mathbf{N}$, the following statements are equivalent :
(i) *There exists a finite, solvable group G and non-isomorphic transitive G-sets of cardinality n which are not linearly equivalent.*
(ii) *There are primes p, q, r with $pqr|n$ and $q|p(p-1)$.*

6.2 *Deciding if $P \in \mathbf{Z}[X]$ has a root mod every integer*

A necessary condition for a polynomial $P \in \mathbf{Z}[X]$ to have a root in $\mathbf{Z}$ is that it has a root mod m for every m. Unfortunately, this is not a sufficient condition.

For example, it can be seen as a simple application of the quadratic reciprocity law that $(X^2 - 13)(X^2 - 17)(X^2 - 221)$ has roots mod m for every m but, evidently, has no rational root.

What if $P \in \mathbf{Z}[X]$ is irreducible ? Then, a deep theorem of Chebotarev implies that a similar cannot happen now. In fact, one has the effective result that given any irreducible P of degree > 1, there is a computable constant $N > 0$ such that for some prime $p \leq N$, P has no root mod p. The following beautiful result has been proved by Berend & Bilu ([2]).

Let P be monic and $P = h_1 \cdots h_r$ be its factorisation in $\mathbf{Z}[X]$ into monics. Let L be $Spl_{\mathbf{Q}}(P)$, and $G = Gal(L/\mathbf{Q})$. For each h_i, fix a root θ_i and let $K_i = \mathbf{Q}(\theta_i)$. Put $H_i = Gal(L/K_i)$.

Three more constants depending on P which will feature, are the following.

Write $\delta = \prod_{i=1}^{r} Res(h_i, h_i') = \prod_{j=1}^{s} p_j^{m_j}$,

$\Delta = \prod_{j=1}^{s} p_j^{2m_j+1}$ and $D = (\prod_{i=1}^{r} \delta_i^{1-1/n_i})^{n_1! \cdots n_r!}$ where $n_i = deg(h_i)$.

Then, we have :

Theorem ([1]**)**

The following are equivalent :

(i) *P has a root mod every $m \in \mathbf{N}$;*

(ii) *P has a root mod Δ and*

$$\bigcup_{g \in G} \bigcup_{i=1}^{r} g H_i g^{-1} = G \ ;$$

(iii) *P has a root mod Δ and mod every prime $p \leq 2D^c$, where c is an effectively computable constant.*

Suggested Reading :

1. D.Berend & Y.Bilu, *Polynomials with roots modulo every integer*, Proc.Amer.Math.Soc. 124 (1996) P.1663-1671.

2. H.Cohen, *A course in computational algebraic number theory*, Springer-Verlag 1996.

3. B.De Smit & H.W.Lenstra,Jr., *Linearly equivalent actions of solvable groups*, J.Algebra 228 (2000) P.270-285.

4. Susan Landau, *How to tangle with a nested radical*, Math. Intelligencer, Vol. 16 (1994) P.49-55.

5. H.W.Lenstra, Jr., *Algorithms in algebraic number theory*, Bulletin of the AMS Vol. 26 (1992) P.211-244.

6. References in 5.